DOCUMENT RESUME

ED 023 606 24 SE 005 582

By-Richardson, Jesse O.

Teaching Mathematics Through the use of a Time-Shared Computer. Final Report.

Massachusetts State Dept. of Education, Boston.

Spons Agency-Office of Education (DHEW), Washington, D.C. Bureau of Research.

Report No-CRP-H-212

Bureau No-BR-5-0311

Pub Date Mar 68

Contract-OEC-5-10-320

Note-224p.

EDRS Price MF-$1.00 HC-$11.30

Descriptors-*Computer Oriented Programs, Computers, Curriculum Development, *Elementary School Mathematics, *Instruction, *Mathematics, Program Development, *Secondary School Mathematics

Reported is a project designed to use the computer as the basis for a laboratory approach to the presentation of mathematics. Classroom instruction was augmented by student experiments in devising and testing mathematical algorithms on the computer. Research procedures for grades 6 through 12 focused on the following kinds of problems: (1) programming a time-shared computer to serve as a useful tool for teaching mathematics, (2) teaching classroom teachers the necessary techniques for using this tool successfully, (3) developing multiple-user computers on an economical basis for school use, and (4) augmenting the mathematics curriculum to make effective use of the computer as a tool for classroom instruction. The research revealed that it is possible to construct programming languages that can be effectively taught to elementary school children. Children are easily motivated to write programs at consoles, and programming work can facilitate the acquisition of rigorous thinking and expression. Such mathematical concepts as variable, equation, function, and algorithm can be presented with clarity in the context of programming. (RP)

BR-5-0311
PA-24

Final Report

Project No. 5-0311 (H-212)
Contract No. OE-5-10-320
PA-24

TEACHING MATHEMATICS THROUGH THE USE OF A TIME-SHARED COMPUTER

March 1968

U. S. DEPARTMENT OF
HEALTH, EDUCATION, AND WELFARE

Office of Education
Bureau of Research

Final Report

Project No. 5-0311 (H-212)
Contract No. OE-5-10-320

TEACHING MATHEMATICS THROUGH THE USE OF A TIME-SHARED COMPUTER

Jesse O. Richardson

Commonwealth of Massachusetts
Department of Education

Boston, Massachusetts

March 1968

The research reported herein was performed pursuant to a contract with the Office of Education, U. S. Department of Health, Education, and Welfare. Contractors undertaking such projects under Government sponsorship are encouraged to express freely their professional judgment in the conduct of the project. Points of view or opinions stated do not, therefore, necessarily represent official Office of Education position or policy.

U. S. DEPARTMENT OF
HEALTH, EDUCATION, AND WELFARE

Office of Education
Bureau of Research

3.0 TABLE OF CONTENTS

3.1 ACKNOWLEDGEMENTS

This report of the research program conducted under Curriculum Improvement Project 5-0311 (H-212), was prepared as a joint effort of the Massachusetts Department of Education and Bolt Beranek and Newman, Inc., (BBN), of Cambridge, Massachusetts.

The remote terminals used in the participating schools, the multiple-user realrime, computer facilities and associated software were provided for Project H-212 by BBN, which furnished consultative and staff services in scientific collaboration with the Massachusetts Department of Education.

Mr. Wallace Feurzeig, Director of the Educational Technoloty Division for BBN, and technical coordinator for Project H-212, provided major assistance in the compilation of the body of the report and related documentation of project activities.

4.0 SUMMARY

The design of Project H-212 envisaged making use of the computer as the basis for a laboratory approach to the presentation of mathematics. Classroom instruction was to be augmented by student experiments in devising and testing mathematical algorithms on the computer. The following problems were identified for investigation at grades 6 through 12:

1. how can a time-shared computer be programmed as a useful tool for teaching mathematics

2. how can classroom teachers be taught the necessary techniques to enable them to use this tool successfully

3. how can multiple-user computer facilities be developed on an economically feasible basis for school use

4. how can the mathematics curriculum best be augmented to make effective use of the computer as a tool for classroom instruction

The students participating in Project H-212 were taught to program a computer--or, more precisely, to write programs in a particular programming language. The primary goal was not to impart facility in programming for its own sake, but rather to exploit it for the presentation of mathematical ideas through classroom instruction and individual student laboratory work.

The work performed in Project H-212 sought to show that the teaching of the set of concepts related to computing, programming, and information processing could be used to facilitate and enhance the presentation of standard school mathematical curricular material, including arithmetic, algebra, and elementary calculus. Two new technological developments contributed significantly to the project effort. New mathematical languages had been developed to make computer programming very much easier to learn and to use in mathematical work. Also, time-shared multi-access computer systems with remote communication consoles were available, making the widespread use of computers in schools economically feasible. From the work carried out by the project at grades 6 through 12, the following conclusions are drawn:

(1) It is possible to construct programming languages of great expressive power yet so simple to learn that they can be effectively taught to elementary school children.

(2) Children are easily motivated to write programs at computer consoles. This kind of mathematical activity is immensely enjoyable to children generally, including those not in the top levels of mathematical ability.

(3) Programming work facilitates the acquisition of rigorous thinking and expression. Children impose the need for precision on themselves through attempting to make the computer understand and perform their algorithms.

(4) A series of key mathematical concepts such as variable, equation, function, and algorithm, can be presented with exceptional clarity in the context of programming.

(5) The use of a programming language effectively provides a working vocabulary, an experimental approach, and a set of experiences for discussing mathematics. Mathematical discussion among high school students, relatively rare in the conventional classroom, was commonplace in this laboratory setting.

(6) Computers and programming languages can be readily used in either of two ways in the mathematics classroom,
   (a) By individual students for independent study on extracurricular problems or special projects.
   (b) As a laboratory facility to supplement regular classroom lecture and discussion work. In this mode students are given assigned problems to work out at the computer.

(7) A third way of using computers and programming, that might have radical implications for the presentation of mathematics, was uncovered - the concept of teaching programming languages as a conceptual and operational framework for the teaching of mathematics.

5.0 Introduction

There are at least two distinct kinds of innovation possible in mathematics curricula. The first of these has to do with the introduction of new mathematical content for its own sake; the second with improving the acquisition of mathematical thinking irrespective of the specific mathematical content. The two may, of course, overlap. Thus we might argue that it is a good thing to teach elementary group theory in high school on the ground that the concept of group is so very important, not only in contemporary physics and contemporary mathematics but in many other disciplines influenced by contemporary abstract mathematical thinking. Seen in this light, the teaching of groups is an innovation of the first kind. On the other hand, it might also be argued that teaching group theory is a better way of leading students towards a mathematical way of thinking than teaching trigonometry. If we stress this aspect, we are presenting it as an innovation of the second kind.

The teaching of the set of concepts related to computing, programming, and information processing could easily be supported by arguments of the first kind. Indeed, the introduction of an elective class in computing in the eleventh or twelfth grades for students with special mathematical ability and interest is no longer unusual. Our intention in the work performed in Curriculum Improvement Project H-212, however, was of the other sort: we sought to show that these concepts could be used to facilitate and enhance the presentation of standard school mathematical material including arithmetic, algebra, and elementary calculus.

The use of computers to administer programmed drills and lessons (this is usually what is meant by *computer-assisted instruction*) has been under investigation in mathematical applications for some years. In this mode of use, though the student uses a computer, he needs to know no more about programming than the television viewer does about electronics. The approach to using computers taken in Project H-212 was very different.

Two technological developments successfully demonstrated during 1962-1964 were instrumental in enabling the work done in this project. New mathematical languages were developed to make computer programming very much easier to learn and to use in mathematical work. Also, time-shared multi-access computer systems with remote communication consoles were realized, making the widespread use of computers in schools an economically feasible prospect. We realized that the new languages could readily be adapted for use by students, including even elementary school children. We believed that teaching students to program problems on a computer could could provide a means for elucidating the issues which cause the greatest difficulties for students of school mathematics.

Specifically, we envisaged using the computer as the basis for a laboratory approach to the presentation of mathematics. The classroom lecture and discussion would be augmented by student experiments in devising and testing mathematical algorithms on the computer. In working toward the elaboration of this idea, we defined the following problems:

1. how can a time-shared computer be programmed as a useful tool for teaching mathematics

2. how can classroom teachers be taught the necessary techniques to enable them to use this tool successfully

3. how can multiple-user computer facilities be developed on an economically feasible basis for school use

4. how can the mathematics curriculum best be augmented to make effective use of the computer as a tool for classroom instruction.

The students participating in Project H-212 were taught to program a computer--or, more precisely, to write programs in a particular *programming language*. The primary goal was not to impart facility in programming for its own sake, but rather to exploit it for the presentation of mathematical ideas through classroom instruction and individual student laboratory work.

The use of one of those objects known as an algorithmic programming language can be seen in a dual aspect. On the one hand, it is an instrument for controlling a computer. On the other hand, it is a formal system specially designed for the expression of algorithms. Our conceptual emphasis is on the latter aspect but we can exploit the former to concretize and motivate ideas which to a young mind might seem highly abstract and difficult. There are several ways in which the use of programming languages on computers appeared valuable in this connection.

1) A major obstacle to learning mathematics is the difficulty in understanding and seeing the necessity for the notion of precise formal literal statement. Up to the time of the initiation into mathematical work, the only mode of expression that children have known is one that presupposes a listener who is able to make reasonable interpretations of what they say. For the first time they are asked to adopt a language which expresses exactly what they say and not merely something which a reasonable person would understand; it is not surprising that they resist. The value of rigor in mathematical thinking needs no justification here. In this context the literal-mindedness of computers becomes a virtue. To learn to use a computer is to learn to express oneself with precision.

2) Intimately related to the notion of precise formal expression is the cluster of notions including formal implication, formal demonstration, algorithmic manipulation of rules, etc. It is notoriously difficult for children, and even for nonmathematical adults, to understand clearly the idea of passing from a given set of premises to a given conclusion using only a limited, precisely-stated set of rules (as distinct from merely justifying the conclusion using other tacit premises and other tacit rules of inference). Students see the issue in a more concrete and in a more personal way through attempting to make the computer draw the conclusion than they do by being forced into making explicit steps of their reasoning, the rules used, and so on.

3) A number of relations (such as that of variable to value, of name to thing named, of equation to solution, etc.) acquire a vivid directness in the context of computer operation. Some modern teachers of mathematics try to render these abstract relations material by such devices as thinking of the variable "x" in classical algebra as a box in which a number can be placed or written, or as a slate on which it can be drawn. The use of the computer not only goes much farther in this direction, but does so in a way that permits more sophisticated manipulations than can be carried out in a natural way on boxes and slates.

4) Many children do not understand how to pass from an English description of a relation or procedure or problem to a symbolic representation of the same thing. This difficulty is dramatically manifested when they are confronted with problems (such as algebra word problems) that do not have obvious mechanical solutions. An understanding of symbolic representation can be developed easily and naturally through the use of computers. The activity of programming provides a natural context for describing mathematical objects and processes.

5) The concept of program leads very naturally to a greater unity in the notion of algorithm. A formula πR^2, or 1/2 BxH is an algorithm, and writing a program to compute the area of a circle or a triangle is in some sense much the same thing as writing down a formula to express the same computation. But not all algorithms used in elementary algebra are formulae in this sense. Thus, finding the solution to an equation by what is called trial and error, or by systematic search, or trying to factor a polynomial by some kind of trial and search procedure does not always appear to students as algorithmic, even when it is. Writing a program to carry out operations of this sort unifies algorithmic processes under one concept and shows concretely that they are not algorithmic when they are not.

6) A large part of learning mathematics comes from doing mathematics. When students are frustrated by a problem, a concept, or an algorithm, they should be encouraged to operate with it so as to better understand and express their difficulties. An algorithm, when represented as a program on a computer, becomes accessible to students, even when they don't fully understand it. Programs are responsive objects. They can be "run"; they can be tested; they can be "fixed"; they can be simplified, extended, or otherwise modified. Students readily learn how to check and revise programs. In so doing they gain skill and confidence in their ability to do mathematics.

6.0 Methods

The project was planned to be conducted in three phases:

1. An in-school pilot phase during the spring of 1965 to install and test equipment, to familiarize participating teachers with the use of the programming language and the operation of the computing system, to carry out limited teaching experiments, to design tentative instructional materials and procedures, and to establish evaluation procedures.

2. A summer study session involving educators and members of the project advisory board, to report on results of the pilot phase, to provide training for additional teachers involved in the program, and to review and improve the curriculum materials for the third phase.

3. The experimental phase to be conducted during the 1965-1966 school year. A detailed two-way experimental design with extensive pre-test and post-test measures was planned.

The experimental phase was to be carried out in five schools with students at three grade levels - 6, 9, and 11. The computer was to serve the students as a mathematics laboratory facility in the following fashion. A mathematics student, sitting in a classroom or a laboratory, works directly at a teletypewriter that is connected, through telephone lines, to a digital computer located several miles away. Simultaneously, other students in other schools are using teletypewriters connected to the same computer. The students, using a programming language, attempt to solve mathematical problems represented as programs. The laboratory work was to be coordinated with the classroom lecture and discussion.

We planned to demonstrate the economic feasibility of such time-sharing techniques for use in supplementing the presentation of school mathematics. We planned also to test experimental hypotheses to the effect that the use of an appropriate programming language on a time-shared computer

would show marked motivational and conceptual benefits in the performance of mathematics students.

The project was actually initiated in June 1965. Thus, the pilot phase activities, scheduled for the spring of 1965, had to be incorporated for the most part in the six-week summer study phase. This shift resulted in a severe reduction in the time available for designing curriculum materials and instructional procedures. Consequently, the controlled experiment planned for the third phase could not be conducted. Instead, we designed a series of methodological explorations at each of the participating schools, to investigate different modes of incorporating programming and computation concepts into the mathematics curriculum. The project activities, as implemented, are described in the remainder of this section. A detailed project history, including a description of the original experimental design and the administrative arrangements, is attached as Appendix 1.

6.1. Summer Study

The summer study phase was conducted during a six-week interval in June and July, 1965. The firm of Bolt, Beranek, and Newman, Inc., (BBN) was designated by the Massachusetts Department of Education as a subcontractor to provide the computer facilities used during the summer study and the subsequent experimental phase. BBN provided the personnel and classroom facilities for training the participating teachers in programming. BBN also provided consultative services in the use of computers in mathematics education.

The notion of teaching a simple programming language to students, and of using this process as the basis for a <u>mathematical laboratory</u> to augment mathematics instruction was conceived by the project personnel just prior to the summer study phase. The programming language chosen,TELCOMP, is described in Section 6.2.1 and in Appendix V. The 16 school systems and 27 mathematics teachers participating in the summer study are listed in Appendix VI.

About half of the study phase was spent in training the teachers to the point of imparting fluency in the use of TELCOMP** This training was done through lecture, discussion, and demonstration supplemented by considerable programming work at the computer terminals. The practical phase of the instruction was carried out in groups composed of four or five teachers, working together on assigned programming problems.

The last part of the summer study was dedicated to planning the various ways in which the separate schools expected to use TELCOMP to augment mathematics instruction during the upcoming academic year. Mainly this involved designing mathematics problems and associated TELCOMP programs,to be assigned to students or to be used as demonstrations. This work was done in teams representing the grade levels primarily involved - 6, 9, and 11 - and in part by individual school teams comprising the teachers in the same school or school system, to take into account the various aspects of their individual mathematics curricula.

** Copyright, Bolt Beranek and Newman, Inc.

The 9th grade and 11th grade mathematics classes were, in most cases, courses in contemporary Algebra I and II, respectively. In order to provide the reader with information concerning the topics covered, we next list the outlines of topics covered by these courses, and for a 6th grade elementary mathematics program for one of the participating schools. Problems associated with the majority of the topics listed were treated through the use of the computer.

6.1.1 Outline of 6th Grade Course Topics in Mathematics; Belmont Public Schools - 1965.

I. Place Values and Number Bases

II. Whole Numbers

A. Concepts and relationships between operations

B. Basic Principles

C. Function concept - Including composite function

III. Computing

IV. Number Theory

A. Factors and Factor Trees

B. Multiples

C. Prime numbers and prime factorization

D. Greatest common factor and least common multiple

V. Fractions and Rationals

A. Number Pairs

B. Number Lines

C. Problems

D. Multiplication and Division

E. Multiplication and Division in decimal notation

VI. Geometry

A. Introduction to abstract nature of geometry

B. Congruence of segments, angles, and triangles

C. Properties of congruent triangle

D. Parallel lines and parallelograms

E. Concept of space geometry

VII. Ratio

VIII. Decimals

IX. Percent

X. Integers

A. Negative Integers

B. Basic Operations

XI. Graphing

A. Number Line

B. Coordinate System

C. Graphs of functions

XII. Measurement

A. Length and perimeter

B. Area and surface area

C. Volume

D. Angles

E. Scale drawings

F. Area of triangles

G. Pythagorian Theorem

H. Metric System

I. Circumference of a circle

J. Area of Circle

6.1.2 Outline of 9th Grade Algebra I Course Topics at Phillips Academy - 1965.

FALL TERM:

1. Use of letters for numbers, translations from English to Algebra

2. Formulas, Equations, Word problems with Positive Rationals

3. Number Theory - odd-even, divisibility-primality

4. Axioms for Positive Rationals

5. Negative Numbers

   a) Elementary operations

   b) Distributive axiom - parentheses

   c) Equations & Word Problems

   d) Axioms, some elementary theorems

6. Polynomials in one variable

   a) Elementary operations, Exponentiation

   b) Factoring

7. Polynomials in two variables

   a) Elementary operations, Exponentiation

   b) Factoring

WINTER TERM

1. Sets - examples, intersection, union

2. Real Numbers - decimal representation, approximations

3. Graphs

   a) Linear Equations - point-slope, two-point

   b) Linear inequalities

4. Systems of equations - two and three

SPRING TERM

1. Algebraic fractions
   a) Elementary operations
   b) Equations, inequalities, word problems
2. Radicals
   a) Definition, theorems
   b) Manipulation using arithmetic operations
3. Quadratics
   a) Equations, word problems
   b) Inequalities
4. Trigonometric ratios

6.1.3 Outline of 11th Grade Mathematics Curriculum; Westwood High School - 1965.

I. Sets of Numbers - Axioms for real numbers

II. Open sentences in one variable

III. Systems of linear open sentences:
Linear equations and their graphs
Linear inequalities

IV. Polynomials and factoring

V. Rational numbers and expressions:
Laws of exponents
Operations with fractions
Decimals, scientific notation

VI. Relations and functions:
Linear functions and relations
Quadratic functions and relations

VII. Irrational numbers and quadratic equations:
Real roots of real numbers
Radicals

VIII. Quadratic relations and systems:
Coordinates and distance in a plane
Graphing quadratic relations
Quadratic systems

IX. Exponential functions and logarithms

X. Trigonometric functions and complex numbers

XI. Trigonometric identities and formulas:
Identities involving one angle
Identities involving two angles
Triangle applications

XII. Circular functions and their inverses:
Variations and graphs
Inverse functions and graphs
Open sentences

XIII. Progressions and binomial expansions

XIV. Polynomial functions:
Polynomial functions over complex numbers
Polynomials with real coefficients

XV. Matrices and determinants

XVI. Permutations, combinations, and probability

6.1.4 Planning Computer Programs for the Experimental Phase

Several hundred specific examples of computer programs for use with these topics were developed during the summer study. To illustrate the variety of these examples we list some of the programs written by one of the study groups - that concerned with the 11th grade (Algebra II) curriculum. We include lists of programs designed as classroom demonstrations, as well as those to be written by students as assigned problems. The rationale of the members of this study group is as follows:

> "This is a group of programs which will be used in the teaching and learning of the material ordinarily considered as Algebra II. These programs are of two types: (1) for demonstration use by teachers, and (2) for assignment to students."

> "The teacher demonstration programs have been included for a number of reasons. First, much class time which would otherwise be spent at the blackboard could be saved when proper programs are available. For example, a demonstration which can show the _y_ value of a hyperbola and its asymptote for a sequence of increasing values of _x_ is often attempted by teachers. The arithmetic becomes involved, time is consumed, and errors frequently occur. One small error can destroy the whole effect as well as waste time. Therefore, we feel that conclusions can be drawn not only in less time but with the greater force which guaranteed accuracy will insure. Also, such a program can be made available to students to assist in assignments. They will thus be able to perform relatively long calculations without becoming involved in tedious manipulations. Homework is almost always designed to illustrate one or more points which the teacher perceives as useful to a basic understanding. With the aid of the computer more illustrative problems can be performed. Moreover, the example assigned can be realistic since we would no longer be bound to the standard textbook examples."

"Perhaps an even more important, though less tangible, reason for such programs is the stimulation of the student's curiosity. If he has a fairly powerful program on such a topic as quadratic equations, it is not unreasonable to hope that he will be motivated to experiment with different co-efficients to determine their effect on the roots, axis of symmetry or extreme point. Although this might not be a universal outcome, some students will be able to learn by discovery."

"The teacher demonstration programs consist of a high degree of mechanical difficulty. We wish to avoid as much as possible the intrusion of programming *per se* on the mathematical content of the course. We believe that pre-programmed teacher demonstrations will allow the students to concentrate on mathematical concepts."

"Most students will accept more readily ideas which have inductive support. Perhaps after deriving the equation of an ellipse from the focal radii definition, the concept can be reinforced if we consider various ellipses and actually show that the sums of the focal radii are equal."

"Also appended is a list of topics which the students will program and execute. If the understanding of a process is involved, we feel that the student will strengthen his knowledge by actually writing the series of instructions for the computer. One example of such a process is finding a square root by Newton's method. Usually we teach this by demonstrating the reasoning and then requiring the student to do a number of examples. A detrimental effect of this method is that in doing this work the understanding often becomes obscured by the arithmetic. If the student is asked to prepare the appropriate series of instructions for a computer, he must master and organize the method in his mind. It is axiomatic that one never learns a topic well until he teaches it; in this project we are asking the student to teach the computer."

"Another extremely important advantage of student programming relates to the "general case". For example, one assignment might be to write a program which would solve two linear equations in two unknowns. The student program must allow for the dependent and inconsistent cases. This and other programs of a similar nature force the student to consider topics in a more general mode than he had been required to do previously."

List of Topics for the Teacher to Program:

Criteria for Teacher Programming

1. Minimizes computation time in class.

2. Increases plausibility of abstract mathematical concepts.

3. Permits the use of output information while avoiding non-relevent or cumbersome student programming.

4. Motivates and challenges the student to a broader study of mathematics.

Demonstration Programs:

1. Finding the prime divisors of any number

2. Generating any number of consecutive prime numbers.

3. Showing the density of rational numbers.

4. Showing the effect of a,b, and c on the graph of $y=ax^2 + bx + c$.

5. Finding points of the graphs of $y=2^x$ and $y=\log_2 x$.

6. Finding points of the graphs of $y=10^x$ and $y=\log x$.

7. Showing the convergence of a hyperbola to its asymptote.

8. Comparing synthetic division with x-a division to f(a).

9. Showing the limit of an infinite geometric series with $|r|<1$.

10. Simplified versus unsimplified calculation of exponent problems. (this program demonstrates the importance of a knowledge of the laws of exponents and also round-off errors in unsimplified calculations.)

11. Changing fractions to repeating decimals.

12. Generating the locus of points for any conic.

13. Showing π and e as the limit of an infinite series.

14. Factoring the general quadratic trinomial.

15. Graphing the trigonometric functions.

16. Solving a triangle given any three parts.

17. Showing Sine and Cosine as the limit of an infinite series.

18. Interpolation of trigonometric and exponential problems.

List of Topics for the Student to Program:

Criteria for Student Programming:

1. A more thorough understanding is accomplished.

2. Repetitive computation is reduced.

3. Increased interest in class and independent work is generated.

Student Programs:

1. Testing properties of inequalities.

2. Testing group properties using $a \cdot b = \max(a,b)$ on the integers 0-9.

3. The greatest integer function.

4. Absolute value.

5. Verify the triangle inequality theorems, $|x+y| \leq |x| + |y|$, etc.

6. Finding the GCD and LCM of two numbers.

7. Solving $ax+b=c$.

8. Solving $ax+b > c$.

9. Testing $ax+by>c$.

10. Prime factorization

11. Least common multiple.

12. Greatest common divisor

13. Finding the square root of a number by the guess method.

14. Simultaneous equations without determinants.

15. Problems with fractional exponents.

16. The distance formula

17. The slope formula

18. The midpoint formula

19. The nature of the roots of a quadratic using the discriminant.

20. The quadratic formula

21. Write the equation of a quadratic given the roots.

22. Determine the type of conic given $ax^2+by^2+cx+dy+e=0$

23. Sum of the focal radii of an ellipse.

24. Finding the center, foci, vertices, major and minor axes of an ellipse.

25. The difference of the focal radii of a hyperbola.

26. Finding the center, foci, vertices, conjugate and transverse axes of a hyperbola.

27. Arithmetic progressions.

28. Geometric progressions.

29. Finding the coefficient of the rth term of the binomial expansion:

(a) factorial method
(b) recursive method
(c) combinations

30. Graphing basic trigonometric functions with variations,

31. Isolating real roots between two successive integers.

32. Finding real roots of an equation,

(a) Newton's method without derivatives
(b) Iteration
(c) Approximation

33. Computation using the law of Sines

34. Factorials

35. Combination of n things taken r at a time.

36. Permutation of n things taken r at a time.

37. Reducing powers of 1.

38. Verifying log properties.

39. Changing bases in logs.

40. Converting degrees to radians.

41. Converting radians to degrees.

42. Verify identies and checking solutions to trigonometric equations.

43. Reduction of trigonometric functions of positive angles.

44. Computation of problems using the law of Cosines.

45. Finding roots of complex numbers using DeMoivre's theorem.

Many such programs were specified for use with topics in applied methematics, in 12th grade advanced mathematics, and in various high school science courses. As an example, the following is a partial listing of programs that were designed to be written by students of chemistry.

A LIST OF SOME COMPUTER PROGRAMS FOR CHEMISTRY - LEXINGTON HIGH SCHOOL - 1965

1. Conversions

   A. Temperature Conversion

   B. Conversion of English to Metric Units

   C. Conversion of Velocity Units

2. Computation - Problem Solving

   A. A Measurement and Its Uncertainty

   B. Determination of Avogadro's Number

   C. The Bohr Atom; Orbital Radius

   D. The Freezing Point Constant of Water

   E. The Equilibrium Constant of Two Chemical Reactions

   F. Gas Law Problems

   G. A Problem in Simple Harmonic Motion

3. Chemical Calculations

   A. Finding the Percentage Composition of $KClO_3$

   B. A Weight Problem for The Chemical Reaction Between Iron and Sulfur.

4. Simple Programmed Instruction

   A. Instruction in Learning pH Relationships

   B. Instruction in Finding Molecular Weight, Percentage Composition, and Mole Amounts

6.2 An Introduction to Telcomp Programming

We submit that the activity of writing or composing a computer program is relevant to the learning of mathematics. To show this, we must give a brief exposition of the main features of the programming language - Telcomp - that was used in the experiment. The discussion should make plausible the fact that elementary school children can easily learn programming. We shall show in subsequent sections how programming work was used in the mathematics classroom during the term of the research project.

6.2.1 Telcomp Expressions and Operations

In Telcomp one manipulates numerical expressions. A Telcomp expression can be an integer such as 376, a decimal number such as -74.295, or a numerically-valued variable. Variables are designated by English letter strings such as X or VOLUME. Subscripted variables or arrays are designated by setting off the indices with brackets (or parentheses) as follows: X[7] or VOLUME [X,Y,Z].

Several types of operations can be performed on these objects. These include the usual arithmetic operations of addition, subtraction, multiplication, division, and exponentiation which are designated, respectively, by +, -, *, /, and ↑. To perform these operations on the computer terminal (which is a teletype console similar to an ordinary typewriter), one uses the TYPE operation. Thus, if one types TYPE 2+3 and presses a key that indicates the end of input, Telcomp advances the paper one line and then types back:

    2+3=5.

It then advances the paper another line and types the mark ← which indicates that it is ready for another input.

Similarly, the following inputs (shown on the left) will cause Telcomp to make the corresponding outputs (shown on the right).

Input	Telcomp Output
← TYPE 2-3	2-3 = -1
← TYPE 2*3	2*3 = 6
← TYPE 2/3	2/3 = .66666667
← TYPE 2↑3	2↑3 = 8

These operations may be compounded, and parentheses can be used to indicate grouping. Thus, TYPE 2.75+3↑(2-3+4) causes Telcomp to type 2.75+3↑(2-3+4) = 29.75. Several operations can be requested on a single input. Thus, the input TYPE 2-3, 2*3, 2/3, 2↑3 will result in all of the outputs shown above.

Variables are assigned values by the SET operation (LET is equivalent), as follows:

```
    Input                     Telcomp Output
  ← SET X=4                   (No output)
  ← TYPE X*(X-1)              X*(X-1) = 12
  ← SET Y=X 2,A=3*Y           (No output)
  ← TYPE X,Y,A                X=4,Y=16,A=48
  ← SET X=1+X  (This means replace the value of X by 1
               plus its current value.  The equal sign
               in a SET command denotes replacement.)
  ← TYPE X                    X = 5
  ← SET LENGTH = 5, WIDTH = 17.1
  ← SET AREA = LENGTH * WIDTH
  ← TYPE AREA                 AREA = 85.5
```

As well as arithmetic operations, Telcomp expressions can include functions such as square root, exponential, natural logarithm, logarithm (base 10), sin, cos, etc., etc.. The following Telcomp instructions illustrate the use of the first three of these functions.

```
← SET X=6
← TYPE X,SQRT(X+1),EXP(X),LN(2.7),EXP(LN(X))

    X=6
    SQRT(X+1) = 2.645751        (Telcomp
    EXP(X) = 4Ø3.4287           output)
    LN(2.7) = .9932518
    EXP(LN(X)) = 6
```

The last example shows how functions can be combined in Telcomp.

The functions MAX(A,B,...) and MIN(A,B,...) determine the maximum and the minimum of the values of the arguments enclosed in parentheses. The functions IP(A) and FP(A) evaluate the integer part and fractional part of the argument, respectively. Thus,

```
← SET V=MAX(3↑2,SQRT(1Ø3),12.423)
← TYPE V, IP(V), FP(V)
```

causes Telcomp to type

```
V = 12.423
IP(V) = 12
FP(V) = .423
```

Telcomp also understands the conditional operation IF as a statement modifier. It interprets the relations <, =, > in the usual way as less than, equal to, and greater than, respectively. Thus,

```
← SET X=2, Y=3
← TYPE X↑Y IF Y>X
← TYPE X-Y IF X-Y>Ø
```

causes Telcomp to type

```
  X↑Y = 8
```

It does not type the value of X-Y since X-Y is <u>not</u> positive.

6.2.2 Telcomp Programs

A Telcomp command with a number in front of it is called a STEP. A STEP is numbered with an integer (called a PART number) followed by a decimal. STEPS are grouped together in PARTS. Thus, STEPS 4.1, 4.2, and 4.3 make up PART 4 in the following Telcomp program.

```
← 4.1 SET ZEBRA = A
← 4.2 TYPE B↑(ZEBRA/2)
← 4.3 TO STEP 2.5 IF A<1Ø
```

(The last command is a TO command and directs Telcomp to find its next command at STEP 2.5, whatever that is, if the value of A is less than 10. TO commands are used to change the order in which Telcomp carries out commands.)

Telcomp cannot carry out a program until it is given a DO command. Thus, if A=6, and B=5, the command

```
← DO PART 4
```

will cause steps 4.1, 4.2, and 4.3 to be performed, causing Telcomp to type

```
   B↑(ZEBRA/2) = 125
```

and then to find its next command at STEP 2.5 in PART 2.

One other command can be used to set values of Telcomp variables. The command DEMAND causes a program to wait for

the user to type in a requested value, as in the following example.

```
← DEMAND A
← SET C = A+SQRT(A)
← TYPE C
← DO PART 1
```

The DO command carries out STEP 1.1. This causes Telcomp to type

```
A =
```

and wait for the user to respond. If the user types in 16 (followed by a carriage return), Telcomp will carry out STEP 1.2 and STEP 1.3, causing the typeout

```
C = 2Ø
```

The command modifier, FOR, is used to successively set a variable to each of a range of values. Thus,

```
← TYPE X↑2 FOR X=3,5,7,9
```

causes Telcomp to type

```
X↑2 = 9
X↑2 = 25
X↑2 = 49
X↑2 = 81
```

An equivalent command is

```
← TYPE X↑2 FOR X=3:2:9
```

In this command, the modifier

FOR X=3:2:9 means for X=3 in steps of 2 until X exceeds 9.

The FOR modifier is often used in programs. Thus, the following program, which computes the first Pythagorean triples.

```
← 2.1 SET C = SQRT (A 2 + B 2)
← 2.2 TYPE A,B,C IF FP(C) = Ø
← DO PART 2 FOR A=1:1:B FOR B=1:1:8
```

This program causes Telcomp to type

```
A=3
B=4
C=5

A=6
B=8
C=1Ø
```

The IF condition in STEP 2.2 requires that C be an integer (its fractional part be zero). In this program STEPS 2.1 and 2.2 are executed 36 times corresponding to the (A,B) pairs (1,1), (1,2), (2,2), (1,3), (2,3), (3,3) ... (8,8).

The Telcomp language is a dialect of JOSS, developed at the RAND Corporation to supply programming for scientists and engineers.* BBN programmers adapted Telcomp for the mathematics classroom during the summer study phase. A detailed summary of Telcomp, including its editing and filing facilities, is given in Appendix 5.

6.3 Experimental Phase

During the 1965-1966 academic year, and through much of the following year, computer teletypewriter terminals were installed in eight H-212 schools. There terminals were connected to a large, general-purpose time-shared computer system at BBN† and used by students and teachers on a daily basis. In the case of some elementary schools the computer was used by students from a single class an hour a day. In the case of some high schools the computer was used by students in several classes throughout the day from 9 A.M. to 6 P.M. and often during evenings and weekends. Four of the eight schools were high schools: Brookline, Lexington, Phillips Andover, and Westwood. The other four were elementary schools, one in Winchester, and three in Belmont. The Belmont schools shared a single line to the computer, each school using the computer at different scheduled hours during the day. Other so-called satellite schools in the geographic vicinity of these eight also used the computer as part of H-212, though on a more limited schedule.

Table 1 summarizes the overall school participation in the program. Sometimes the problems assigned to students required several sessions at the computer to work out. To facilitate student work on this extended basis, a facility was incorporated in Telcomp to permit students to file programs that were being worked on or saved for demonstration. Table 2 shows a partial listing of programs on file one day early in 1966. By the end of 1966 there were about 150 programs in the school files on an average day.

---

* Shaw, J.C., "JOSS: A Designer's View of an Experimental On-Line Computing System," AFIPS Conference Proceedings, Fall Joint Computer Conference, Spartan Press, (1964).

† The computer, a Digital Equipment Corporation PDP 1-D, has a time-sharing system designed and programmed at BBN which permits up to 64 terminals to be in operation at any time. Each of the 64 users can be using any of a variety of programming languages, including Telcomp, thus the system is "general-purpose."

Table 1. Outline summary of program participation-1965-1966

School (Town)	Grade	Teacher	Activity
J.S.Kendall Elem. (Belmont)	6	Capron	These classes were to be conducted in accordance with the general experimental design set forth in the proposal. Each teacher, for the grade shown, taught one experimental group and one control group. The sixth grade experimental groups used the computer terminal almost entirely within formal class sessions; the students made negligible use of terminals out of class. The ninth grade groups made only limited use of terminals in class and substantial use out of class, as during study periods, recess, lunch, and after school. Andover use included considerable work during evenings and weekends.
Payson Park " (Belmont)	6	Bixby	
Winn Brook " (Belmont)	6	Conley	
Phillips Academy (Andover)	9	Bedford	
Lexington High	9	Dwyer	
Westwood High	11	O'Malley	
Vinson-Owen (Winchester)	6	Greer	This teacher taught one control and four experimental groups. Students used the terminal both in class and during recess and lunch.
Brookline High	11	Wiggin	This group of Algebra II students used the terminal on an extracurricular basis.
Westwood High	11	Grey	
Westwood High	11	Pender	
Dedham High	9-12	Seeger	Each school in this group used the Westwood terminal one afternoon a week, after school, for 2 hours, on a regularly scheduled basis, for the major part of the entire school year. This student use, though out of class, was part of the regular mathematics program in each case.
Dover-Sherborn Regional High	9-12	Young	
Norwood High	9-12	Buscone	
Walpole High	9-12	Radzwill	
Xaverian High (Westwood)	9-12	Br. Roch	
Lexington High	12	Koetke	This teacher taught a special group of pupils in advanced mathematics heavily based on the use of programming and computational work.

Table 2. Partial listing of programs on school files.

	PHILLIPS ACADEMY
INVERS	COMPUTES INVERSE OF NXN MATRIX
POKER	NOT DEBUGGED
INTAPP	APPROXIMATES INTEGRAL IN FOUR WAYS
PRMDIS	DISTRIBUTION OF PRIMES
LINEQN	NOT DEBUGGED
GEOLIM	APPROX. OF CIRCUM. AND AREA OF CIRCLE
ANGEO1	POINT ABOVE, ON, OR BELOW LINE
ANGEO4	LINE THRU TWO POINTS
ANGEO7	X,Y-INTERCEPTS ND SLOPE
INFBIN	APPROXIMATES INFINITE BIMALS
CIRCLE	SPIRAL GIVING SQUARE ROOTS
	BROOKLINE HIGH SCHOOL
T-T-T	PLAYS TIC-TAC-TOE
FORMAT	FORMAT FOR T-T-T GAME
PLOT	PLOT X, Ø<=X<=3Ø
SYMBLO	DEFINES FNCTNS FOR LOGIC
TRIG	CALCULATES TABLES
STATIC	MEAN, VARIANCE AND CORRELATION
	LEXINGTON HIGH SCHOOL
STEF	SF INEQUALITIES
RACE	BEING WORKED ON
NEWTON	BEING WORKED ON
SOLVE	EL FINDS ZEROES OF A FUNCTION.
LOGIC	GENERATES LOGICAL TRUTH TABLES
RCH	BEING WORKED ON
ICH	LR WK PLEASE CHECK
CHO	ZEROS OF A FUNCTION
FUNC	GUESSES FUNCTION MADE UP BY STUDENT
RICH	BEING WORKED ON
ROULET	PLAYS ROULETTE - (DO PART 1ØØ)
GASJOR	GAUSS-JORDAN REDUCTION
DOXFIN	QUADRATIC INEQUALITIES
CHEMIS	BEING WORKED ON
STEFAN	SF INEQUALITIES
INEQ	SOLVES QUADRATIC INEQUALITIES
HOOP	JS PLAYS BASKETBALL (PART 1Ø)
INEQTY	KJ INEQUALITY PROGRAMME (B.W.O.)
EDLO	BEING WORKED ON
WORK	BEING WORKED ON -- WJK

The administrative operation of the project within a school is illustrated by a report from one of the most active participating schools, Lexington High School.

Report of Lexington High School

Walter Koetke, John C. Dwyer, and Neil Soule

Regularly Scheduled Use:

Terminal is regularly available to any interested students from 7:15 - 5:00 on weekdays and frequently during assorted time periods on weekends and school holidays. The exceptions to this available time are: 1) when terminal has been reserved for a classroom demonstration or use by an experimental class and 2) when computer itself is down. (It should be well noted that Lexington has not lost a single minute of computer use due to Teletype failure.) Computer down time has averaged about 1-1/4 hours per day since the initiation of the project in the schools, hence student use has averaged about 8-1/2 hours per day during the week or 42-1/2 hours per week. To this can be added an average of 3-1/2 hours per week for use on weekends and holidays. A significant problem has been weekend supervision, for the terminal would be used much more extensively during this period if supervision was available.

The average weekly use for each group of students is approximated below:

1)	12th grade scheduled class	4 hrs.
	assigned work	13 hrs.
	unassigned work	7 hrs.
2)	9th grade scheduled classes (2)	4 hrs.
	assigned work	2 hrs.
	unassigned work	1 hr.
3)	10th grade scheduled class	3 hrs.
4)	11th grade chemistry class	1 hr.
5)	Students not in any class scheduled to use computer	8 hrs.
6)	Teachers	2 hrs.
		45 hours.

Student Background

1) 12th grade class - - This is composed of 15 students taking the course "Mathematical Techniques Used in High Speed Data Processing" simultaneously with their sequential 12th grade math course. The mathematical background of these students varies considerably. 4 of the students are also taking a full year of calculus. 5 are studying selected topics of advanced algebra going through differential calculus, and 6 are of average ability following a typical modern sequence of third year algebra.

   This course meets 4 hours a week for a full year. Some of the topics covered are: Computer Programming (TELCOMP, FORTRAN, and a simulated machine language), Boolean Algebra, Logical Circuit Design, Iterative Approximations, Monte Carlo Methods, Linear Programming, etc. Computer use is extensive.

2) 9th grade classes - - Two groups of approximately 30 students each who are studying Algebra II in grade 9. All have received at least a C in 8th grade Algebra I, and hence the group must be regarded as slightly above the average 9th grade ability level. The course uses the text Algebra II, by M. Dolciani. Computer use has been chiefly for demonstration, although all students have been required to write several short programs, and interested students have been allowed to use terminal at their own pace.

3) 10th grade class - - A group of 30 highly selected students following an "Advanced Placement" program. Basic text used is Geometry, by E. E. Moise and F. L. Downs, supplemented by work in logic, Boolean Algebra,

and analytic geometry. Classroom use to date has been limited to a very few demonstrations. (Class has not yet begun the analytic geometry.) The entire class was taught the TELCOMP language in 3 hours, but no assignments were made. Many students explored the terminal use on their own time, most of them were allowed to miss occasional class lectures in order to use the terminal. Some very sophisticated programs were written under these conditions of only being guided by the teacher.

4) 11th grade chemistry class - - An "Advanced Placement" group using the Chem-study texts. This group did not have access to the terminal for in-class use. However, several hours of class time were used to teach the TELCOMP language. They were then free to use the terminal as they saw fit. Most of the students made use of the computer in analyzing the results of lab experiments, and about 5 students explored more sophisticated programming in both chemistry and other areas.

5) Students not in a class scheduled to use computer - - About 40 students with no formal ties to any class even remotely connected with the use of the computer have learned the TELCOMP language and written programs of interest to them. These students have learned by watching and questioning their peers with occasional support from teachers.

The mathematical background of this group and their year in school cover the spectrum of all possibilities. There are 9th graders through 12th graders, and students in "Advanced Placement" courses through students who are consistent underachievers. The common ground of this group was an enthusiasm for working with the terminal, and an individual advancement of knowledge.

Terminal Location and Accessibility

Lexington has very close to an ideal situation for allowing students maximum use of the terminal. The terminal is located in a classroom-office combination which is always occupied by a school secretary and almost always by a teacher as well. This means that at no time during a school day is the terminal unavailable to students due to lack of supervision. A connection for the terminal has also been made in a much larger classroom adjacent to the one described, and the classes which are to use the terminal are scheduled into that room. Hence, by moving the terminal about 40 feet it is in another room where it can be demonstrated to larger groups.

The reason this physical set-up was possible is that Lexington had enough prior notice to enable them to plan this way when the master schedule was made. Such notice, with a set of minimum requirements based upon anticipated use, should be given to all participants far enough in advance so they can plan for optimum use of the computer terminal.

Student Reaction to Terminal Use

Students of all grades and ability levels seem to follow the same three-stage pattern in learning to use the terminal. The more able students pass through the first two stages much more rapidly, but the three stages are always evident.

The first stage is one of fascination with the terminal as a "toy." Few students have ever had a device which remembered their name, the date and the time. Few have used devices which could give immediate mechanical and electrical responses to queries they formulated. Little is accomplished during

this stage other than understanding the on-off procedures and the very simple direct commands. The average student should progress through this stage in 1 or 2 hours of terminal use.

The second stage is one of learning the TELCOMP language and basic techniques of programming. During this stage programs can be written for such things as generating prime numbers, ordering a given series of numbers, supplying co-ordinates of points satisfying a given equation, using Euclid's algorithm to find the greatest common divisor, and converting base 10 numbers to binary form. Little or no new mathematics should be taught at this stage. The emphasis must be on problems which illustrate the use of a variety of programming techniques and all TELCOMP instructions. The goal of the teacher in this stage ought to be to have the students become so familiar with programming techniques that they can eventually use the computer as a tool - not an end in itself. This stage can be done (and was done in Lexington) in about 8 hours of class instruction and about 2 hours of terminal time per student. The terminal time for each student is essential, for only by using the terminal himself can a student learn to de-bug his own program.

The third stage fulfills one of the purposes of having a terminal available for students. This stage is reached when students can use the computer as a tool that might be used in a variety of situations - not all of which need be mathematical. When this stage has been reached, the method of attacking a problem will be the challenging task, and the writing of a program once an algorithm has been found will be almost a routine matter. A student must attain this third stage if he is truly going to gain the full benefit of the terminal's presence. For example, a student is told to write a program to solve a general set of n equations in n unknowns.

The benefit of this task lies in the student's ability to understand the complexities of the problem in the general case, and in his formulating a generalized algorithm. Once this has been done, there is little to be gained in transferring his algorithm into a string of computer instructions - hence he should be capable of doing this with ease or he will lose sight of the real purpose of what he is doing and the program - not the algorithm - will be the important result to him. This writing of instructions is necessary, however, for only in this way can the student verify that his algorithm is correct. The program can then be run on the computer and algorithmic corrections and modifications can be incorporated. The importance of reaching this third stage cannot be overemphasized.

The importance of teaching the students to use the computer as a powerful tool leads to a strong recommendation for making the continuation of this research project more fruitful. Namely, the experimental classes should spend the first two or three weeks learning programming and the TELCOMP language. No pretense should be made that this is part of the usual mathematical sequence. Once this has been done, the computer can be used to its fullest potential for the remaining thirty-eight weeks. With the aid of the computer, the students should better understand many topics in the prescribed sequence and will probably have explored several related topics using the computer. Certainly the two or three weeks lost in the beginning of the year will be more than made up by the year's end.

Student reaction to the terminal has been much more enthusiastic than expected. Lunch period is always very crowded, for many students readily skip eating for just the chance that they might be able to use the terminal. Most students are more than willing to spend 2 hours after school waiting to use

the terminal for 15 or 20 minutes. Third hand rumors that the terminal will be available on a weekend or holiday always brings 5 to 15 students willing to stand outside and hope the rumor is true. There can be no doubt that the computer terminal is one of the best motivators of students from all ability levels. For two years prior to this project, Lexington has offered a course in the programming and use of computers and associated topics from mathematics. We thought student interest was high, but it never came close to the almost uncontained enthusiasm experienced since the terminal was installed. We are presently having students choose their courses for next year, and several hundred are trying to choose a schedule which will either put them in a class scheduled to use the terminal or allow them free time when the terminal is not in a scheduled class.

In spite of extensive student competition for terminal use, there has not been one argument over who uses it and when. There are, of course, some rudiments of a priority system. Use of the terminal for classroom demonstrations has unalterable priority over any individuals, and a student working on an assigned program or project has priority over a student working on a project unrelated to any assigned work in any class. We have allowed students priority over any teacher, thus teachers are often hard pressed to find spare minutes when they and the terminal are simultaneously free. One fascinating and productive facet of this system is the student interaction it fosters. If one student can't get to the terminal before another is finished, he usually tries to help the student using the terminal so he might finish faster. More often than not, they will both get wrapped up in both of their programs - rapidly exchanging ideas, techniques, and approaches. This student to student interaction is perhaps one of the best possible learning situations, and we have observed many significant ideas have been exchanged and "discovered" in this manner.

Types of Applications

The computer was utilized at Lexington for several, significantly differing, purposes. Certain programs were well suited for use in teaching the fundamentals of programming, others served as motivational devices, some were best used for group demonstrations, some as part of an assignment on a particular topic, and still others were entirely self-motivated for a variety of reasons. Although programs discussed in this section are often well suited to more than one use, they will be considered according to their principal use.

Programs Suitable for Teaching Fundamentals of Programming

Topics from elementary number theory offer an abundance of material for demonstrating all types of programming techniques. The mathematical ideas involved in most of these are either already known by 9th grade students, or can quickly be picked up. Even if some teaching of mathematics is required at this stage, the nature of the material is such that most students find it quite interesting and hence learning is rapid. Types of problems included in this area are:

a) Find greatest common divisor and least common multiple of several numbers.

b) Listing coordinates of points which satisfy a given function. (Note: this program can be written in 4 or 5 different ways which, when taken as a group, will demonstrate all TELCOMP instructions and most basic techniques of programming.)

c) Generation of prime numbers, twin primes, and prime triplets.

d) Generation and manipulation of the Fibonacci Series and Pascal's triangle.

e) Factorial function.

f) Base changing.

g) Construction and use of a difference table.

In the solution of these problems one finds it necessary: to use programs containing single and multiple loops; to utilize multiple part programs; to use all types of TELCOMP instructions and most functions which students can understand, to overcome the difficulties caused by machine round-off error; to write flow charts and to develop an algorithmic approach to problem solving.

Because the mathematical ideas are simple, student effort with these is centered about the programs. Students quickly begin an entirely self-motivated competition to write the shortest program to solve a particular problem. This leads them quite naturally to a discussion of the relative merits of program length versus program efficiency -- a topic well worth student exploration. Moreover, this leads also to students' discovering mathematical ideas. For example, one student wrote a prime number generator which was based on the theorem (which the student - a 10th grader - "discovered" and proved) that all prime numbers are multiples of six plus or minus one. His program was both shorter and more efficient than the teacher's -- not an uncommon situation.

Programs as Motivational Devices

A very simple yet effective use of the computer is simply as a tool to motivate a student. This can be done in two quite different ways. The first way is illustrated by a program

which simply requests a number X from the student (suppose 2 is supplied by the student) and then types:

$$X/100 = .02$$
$$\tan(X/100) = .02$$

Certainly this suggests a possible trigonometric identity to most students. This very simple idea, coupled with more subtle problems, has motivated many students to rigorously prove or disprove the validity of the identity on their own.

A second and more involved type of motivational device is best suited for use after students have learned to program well. The simplicity of the TELCOMP language leads most such students to the conviction (frequently correct) that they can write any program anyone else has written. Hence a teacher can create a program and request his student to determine the algorithm being used in the program. This technique has been used on several occasions with excellent results. As an example, one group of students was asked to prove that any three non-collinear points determine a unique parabola. Another group was asked to determine the algorithm being used in a program which would correctly guess the function made up by them for any parabola in standard form after being supplied with the coordinates of any three points which satisfied their function. Except that the second group had TELCOMP programming experience, the ability and background of the two groups was about equal. In the first group there were NO correct solutions, but three-quarters of the second group did find a valid algorithm, and about half of the latter students were able to state and prove the three-point parabola theorem. The results of this particular experiment again seem to demonstrate that appropriate use of a TELCOMP terminal can motivate mathematical understanding by teaching students effective approaches to problem solving.

Programs for Demonstration

Classroom demonstrations of mathematical ideas are useful in many situations, and several teachers have used the computer terminal for brief demonstrations in classes which have no knowledge of programming. The terminal is, however, most effective in demonstrations for classes which are familiar with programming, for these students quickly understand the algorithm being demonstrated and have confidence in their ability to judge the results.

An obvious area for demonstration is probability and statistics. Using the random number generating function, one can easily simulate many probabilistic processes. The computer can be made to toss a coin and report distributions of any desired sequence of results. How many heads and tails after each 100 tosses? How many sequences of 4 heads or 4 tails in a row in each of several series of 100 tosses? The value of pi can be approached by computing the perimeters of inscribed and circumscribed regular polygons for polygons of 10, 20, 30,... sides. Programs like this give students a good feeling for the concept of limits. Most areas of mathematics have many topics well suited for demonstration using the computer terminal. The types and effectiveness of demonstrations are limited only by the teacher's time.

Assigned and Student-Originated Programs

Some of the many problems assigned to students at Lexington are: Gauss-Jordan analysis of simultaneous linear equations, finding the zeroes of a function, solving pairs of non-linear equations, numerical approximations in integral calculus, sentence generation and music writing. As well, many types of chemistry and physics applications have been assigned and diverse problems

in game-playing have been assigned to, as well as originated by, students at all levels.

Let me again stress the point that the purpose of an assigned program should be to have the students themselves devise a general algorithmic solution. The end result is a computer program, but its use is only to check the validity of the algorithm. It should be obvious that there is an abundance of meaningful material for such use.

Report of Lexington High School (9th Grade)

John C. Dwyer

Of all the things that could be said for this first year's use of the time-shared computer system in the classroom, the one outstanding result as far as my ninth-grade classes are concerned is that it has helped them to develop a method of critical analytic thinking that I don't feel they would have gained under ordinary circumstances. They have further developed a genuine appreciation for thought processes and I am convinced that they have improved their mathematical skills. This ability to think a situation through logically has carried over to and caused some improvement in their other subjects as well.

Partly because we have departmental testing, and partly due to parental worry that Johnny, in the computer-assisted course, would get less than Jimmy, in the traditional course, we have not used the terminal as much as we would like. However, now that we have reached the point necessary to satisfy curriculum material, we will attempt to make up for lost time and return to a somewhat fuller use of the terminal during the remaining weeks. Actually, I am not too sure that the lay-off we've

experienced hasn't been beneficial. First, because the students are clamoring to use it again, and secondly, with a fairly good knowledge of algebra through quadratics under their belts, they may make more use of the terminal and be able to explore further some more advanced phases of programming.

In an effort to decide particular uses of the terminal in our own case at Lexington in grade 9, I sat down with my curriculum guide and searched it through to find specific areas where the computer would best fit the needs of my students. In all, there appeared to be some 40-50 topics that seemed to offer the best situations for the use of the computer. After spending some time introducing the students to the machine, the language (TELCOMP) and the fundamentals of programming, we began to use the terminal in a three phase approach to the algebra. First, it was used in the classroom to introduce and demonstrate a particular concept. Secondly, it was used to reinforce an idea after it had been introduced and discussed at some length. Thirdly, it was used as a remedial help. By pre-programming and storing a particular routine, the student could then be sent on his own to work at the terminal until he was successful. At this point let me cite a specific example of this three phase approach: - Linear equations in two variables - Phase 1- after preliminary discussion on the fact that more than one ordered pair of numbers would satisfy the equation, one student sat at the terminal and by solving the equation for one variable in terms of the other he was able to have the computer print out a string of ordered pairs in the form (x, y). Another student was at the overhead projector and using a slide of the Cartesian coordinate system plotted the points as the computer gave them. After repeating this process several times, the idea of the slope of the line was discovered. The two point method was discussed and then the terminal was put to use to

prove that the slope of the line was constant no matter where the points were chosen. From here the discussion led to the discovery of the y-intercept being equal to the constant term with the equation in the $y = mx + b$ form. The assignment for the next time was to graph the solution set of a series of linear equations. A further development along this same topic came when we reached systems of linear equations and the student was required to identify the system as being consistent or inconsistent, dependent or independent. If there was a common solution he was asked to find it. After the preliminaries the students, working in pairs, were assigned the task of writing a program that would identify the system and print the solution if one existed. This example typifies the approach we used in the ninth grade classes.

7. Results and Findings

The computer is often mistakenly thought to be a large, fast slide rule or desk calculator. We found that, with an appropriate programming language, in this case Telcomp, students used it as a tool to facilitate mathematical thinking and problem-solving rather than just numerical calculation. We try next to describe the new kind of laboratory experience that was made possible in this computer context.

7.1 How Students Used Telcomp

The following example, taken from a problem assigned to sixth graders at a Belmont school in 1966, illustrates the kind of interactions with the Telcomp system in which students were routinely involved.

A student is given the problem of determining the least common multiple (LCM) of any three integers (A, B, C). He is given a definition of least common multiple: the LCM is the smallest integer exactly divisible by A, B, and C. He looks at several particular cases and observes how he solves them. Sometimes the solution is obvious, as from "inspection" when the three integers are equal. For some cases he does trial-and-error calculations to gain insight.

At some point he sees that the LCM cannot be less than the largest of the three numbers. Now he can write a Telcomp program for determining the LCM. The program operates as follows: it asks the user for A, B, and C. Then it tentatively sets the LCM to the largest of these three numbers and tests whether the trial LCM is exactly divisible by A, B, and C. If not, it increments the trial LCM by 1 and repeats the test. Otherwise, it types out the value of the LCM. (The program is written here in a schematic form of the Telcomp language often used in sketching out an algorithm before putting it into final form.)

<u>Program 1</u>

```
1.  DEMAND A, B, C
2.  SET TRIAL = LARGEST OF A, B, AND C.
3.  IF TRIAL IS EXACTLY DIVISIBLE BY A, B, AND C THEN
    SET LCM = TRIAL, TYPE LCM, AND STOP.
4.  OTHERWISE, INCREASE TRIAL BY 1 AND GO TO STEP 3.
```

The student then "runs" the program. Outputs from the program follow for three cases. (The student's inputs are underlined.)

```
A = 1         B = 7         C = 1000
LCM = 7000

A = 2         B = 300       C = 5
LCM = 300
```

A = 4 B = 6 C = 1
LCM = 12

The student is sure that the program expresses a valid procedure. Nevertheless, he is unhappy with this program -- it took a long time to get an answer for the first case. He can demonstrate that the LCM cannot be larger than the product of A, B, and C; based on this fact he writes a second program. His new program is similar to his first one, except that it initially sets the trial LCM to the product and successively decreases this trial value by 1 until it becomes exactly divisible by A, B, and C.

Program 2

```
1.  DEMAND A, B, C
2.  SET TRIAL = A X B X C
3.  IF TRIAL IS EXACTLY DIVISIBLE BY A, B, AND C THEN
    SET LCM = TRIAL, TYPE LCM, AND STOP.
4.  OTHERWISE DECREASE TRIAL BY 1 AND GO TO STEP 3.
```

He then runs the second program. He finds that the outputs from the two programs are different for the third case. The new output is:

A = 4 B = 6 C = 1
LCM = 24

Clearly the second program is wrong and he thinks he sees why -- an extra factor of 2 came from A or from B -- but he's not sure how to handle repeated factors, and he would like to have a more elegant demonstration. After some reflection, he sees a new way.

Program 3

```
1.  DEMAND A, B, C
2.  SET MAX = LARGEST OF A, B, AND C.
3.  SET TRIAL = MAX X F WHERE F INCREASES FROM 1 IN
    STEPS OF 1 UNTIL TRIAL IS EXACTLY DIVISIBLE BY A,
    B, AND C.
4.  SET LCM = TRIAL, TYPE LCM, AND STOP.
```

In this program the trial LCM is initially set to the largest of the three numbers (call it MAX). The trial LCM assumes successive values MAX, 2MAX, 3MAX, 4MAX, and so on, until it is exactly divisible by each of the three numbers. This procedure seems to work; nevertheless, the student is not sure that it is general. He cannot prove its validity. When he reports the results of his session, his teacher will clearly see that this sixth grader is ready and motivated to learn about prime number factorization. With such knowledge he will be able to devise a mathematically more relevant solution to the LCM construction.

Although algorithmic languages such as Telcomp can be used simply for performing assigned calculations, e.g., for evaluating formulas, most students in the H-212 project use Telcomp as this sixth grader has. In this way, ideally, a student engages himself in a problem-solving dialogue as he tries to construct an effective procedure. He begins with imperfect conceptions, partially working through his difficulties, and finally arriving at solutions leading to mathematical insight. Even when he is unable to solve the problem, he usually has a clear and definite idea of his difficulties and, once he can express these, he is on his way to a deeper mathematical understanding.

7.2 Three Modes of Use of Telcomp as a Mathematics Laboratory

Programming was used primarily as a laboratory tool to supplement the regular work done in the mathematics class. Students were assigned problems that were to be worked out at the computer. Student programs written during the course of the year illustrate the vast scope of subjects accessible to programming. A partial list of topics in which problems were programmed follows:

Sixth Grade

- Scientific notation
- Decimal equivalents of common fractions
- Order of operations
- Exponential notation
- Equations
- Functions
- Perimeter
- Area
- Volume
- Factoring
- Least Common Multiple
- Equivalent fractions
- Reciprocals
- Decimal notation
- Sieve of Erastosthenes
- Bode's Law

Ninth Grade

- Rational approximation to square roots
- List of primes $\leq n$
- Change a number from base 10 to base B
- Change a number from base A to base B
- Reduce a fraction to lowest terms
- Finding the greatest common divisor and least common multiple of several numbers
- Solving pairs of non-linear equations
- Graphing linear equations
- Solution of simultaneous equations
- Solution of quadratic equations and quadratic inequalities.

Eleventh Grade

- Graphing polynomials
- Sum of first n integers
- Product of first n integers
- Listing coordinates of points which satisfy a given function
- Factorial function
- Three point parabola problem
- Gauss-Jordan analysis of simultaneous linear equations
- Finding the zeroes of a function
- Conic sections
- Analysis of quadratic forms

Twelfth Grade

- Circumference and areas as limits
- Solving triangles
- Series generation of π
- Series generation of sine and cosine
- Series generation of e
- Approximation of integrals
- Value of π by inscribed and circumscribed polygons
- Numerical approximations in integral calculus

Representative examples of problems programmed by students as a part of their course work are included as Appendix 2.

Telcomp programming was incorporated into the school programs in another informal way. Many students carried out independent projects involving either mathematical applications within other parts of the curriculum or extracurricular projects. Some of these projects are listed next.

Scientific and Liberal Arts Applications within Curriculum

- Chemistry problems
- Genetics problems
- Analyzing data
- Sentence generation using random digits
- Analyzing Latin poetry

Extracurricular Projects

- Generation of prime numbers, twin primes and prime triplets
- Generation and manipulation of the Fibonacci series and Pascal's triangle
- Construction and use of difference tables
- Strategic games -- baseball and basketball simulations, poker, black jack, chemin de fer, tic-tac-toe, etc.
- Cryptography
- Photography
- Writing music.

Several independent projects carried out by students are described, along with the associated Telcomp programs, in Appendix 3.

Another distinctly different way of using a computer and a programming language was found and explored in preliminary work. Because the use of the computer had apparently great educational benefits, both conceptual and motivational, the research staff was interested in incorporating programming into the presentation of mathematics in a more integral way than merely supplementing the regular classroom presentation. They claimed that mathematical concepts could be presented in terms of programming concepts. They visualized a mathematics laboratory curriculum in which students use the computer for individual laboratory work, and the teacher in the classroom lecture and discussion develops the mathematical material in terms of programming ideas and experience.

This concept was elaborated in the summer of 1966 and explored in the first part of an introductory algebra course taught at Westwood High School in September 1966. The lesson plans used in the presentation are given in the next section.

7.2.1 A Mathematics Laboratory Course: Lesson Plans for Algebra I

The following pages contain the outlines of the first lessons of a preliminary version of a new sequence of lesson plans for Introductory Algebra. These lesson plans are intended for use at the ninth grade level with students who have had no formal instruction either in algebra or in programming. They provide an introduction to the Telcomp programming language in the context of teaching certain ideas about formal manipulations that are basic to the usual Algebra I course. The material covered is the use of literal expressions, formulas, variables, equations, identities, inequalities. The presentation is based on the idea that an important part of teaching Algebra I is in initiating students into formal ways of thinking.

Subdivision into lessons corresponds to a division of topics rather than a time division of teaching hours. We have little experience on which to base any estimate of how long it would take to teach this material. The presentation is in the form of instruction to teachers. The lessons presuppose access to a computer terminal for motivational rather than purely mathematical reasons. Obviously, programming could be taught without contact with a computer. However, we believe that the need to control the machine will motivate formal and analytic modes of thinking which might seem artificial to students in any other context.

Lesson 1

The first lesson introduces the simplest Telcomp commands and corresponds to the chapters dealing with formulas in algebra textbooks. We begin by considering situations which are familiar to students such as the calculation of the area of a circle. Instead of talking about formulas we talk about procedures. A procedure for calculating the area of a circle is a sequence of steps (operations), already known by students. We now describe the procedure to them. Measure the radius, multiply it by itself, remember the result, and multiply it by π--the resulting number is the area of the circle. Anyone who can carry out these steps can find the area of a circle. Someone who cannot use this procedure might be able to find the area by some other procedure. Our Telcomp computer is able to perform all these steps except the first. To have it calculate the area of a circle, we "tell" it the radius, and it will give us the area. But first, we must describe the procedure in the Telcomp language in a form which the computer can accept via a Teletype keyboard. To understand what this language is like we begin by looking at a procedure written in it.*

Program 1

```
DEMAND RADIUS
SET NUMBER = RADIUS*RADIUS
SET AREA = 3.14*NUMBER
TYPE AREA
```

---

* Although all program steps must be numbered to be accepted by Telcomp, we have omitted these step numbers in these early examples for clarity.

We now go through this program step by step to see what the computer does with it. DEMAND RADIUS is an instruction that causes the computer to type out RADIUS = and then wait for something to be typed in. If a number is typed in, this number is known to the computer, for the while, as RADIUS. The next step contains some peculiarities. The first is the little asterisk. This is nothing but the sign used in this language for multiplication. Thus, RADIUS*RADIUS means RADIUS multiplied by itself or RADIUS squared. Why do we say SET NUMBER = RADIUS*RADIUS? To understand why, go back and look at the original procedure. We said multiply the radius by itself, remember the result, then multiply it by π. The way the computer remembers the result is by giving it a name and using this name in further calculations. So the effect of the instruction SET NUMBER = RADIUS*RADIUS is to take the number that has been called RADIUS, multiply it by itself, keep the result, and give it a name, namely, NUMBER. The next instruction says SET AREA = 3.14*NUMBER. The effect of this instruction is to take NUMBER which is the number obtained in the previous line, multiply it by π, and call it AREA. The last instruction says TYPE AREA and causes the computer to type out the number which has been given the name AREA. This might seem a little confusing, so let's try it on the computer and see exactly what happens. To help us follow the sequence of events, we will add two more instructions. The program which we will put on the computer is the following:

Program 2

```
DEMAND RADIUS
TYPE RADIUS
SET NUMBER = RADIUS*RADIUS
TYPE NUMBER
SET AREA = 3.14*NUMBER
TYPE AREA
```

This program causes the computer to type out RADIUS = . At this point we type in 4. The number 4 now has the name RADIUS. So the next instruction causes the computer to type 4. The next instruction is to multiply RADIUS by itself; that is, to multiply 4 by itself. The result is 16. 16 is now going to be known to the computer as NUMBER, so when it carries out the instruction TYPE NUMBER, it will type 16. It will then multiply NUMBER, i.e., 16, by 3.14 and call the result AREA. AREA is 16 multiplied by 3.14, i.e., 50.24. The last instruction tells it to TYPE AREA, and it will type 50.24. We next consider a program that looks slightly different.

Program 3

```
DEMAND RADIUS
SET FOO = RADIUS*RADIUS
SET BOO = 3.14*FOO
TYPE BOO
```

The teacher should make sure that the class understands very thoroughly that Program 3 has exactly the same effect as Program 1. In general discussion, emphasis should be placed on the fact that numbers can be given arbitrary names and that once they are so named, operations on them can be described without any further thought about the actual numbers involved. The use of mnemonic names as a convenience should be emphasized and contrasted with other requirements such as economy of writing. Thus the same program could be written in yet another form.

Program 4

```
DEMAND R
SET X = R*R
SET Y = 3.14*X
TYPE Y
```

This program has exactly the same effect as the others. It was written with less typing. On the other hand, coming back to it later on, we might have to think for a while to remember what R, X, and Y were.

Exercises on this lesson consist of having the class write other programs of the same general form.

Lesson 2

There is probably a great deal of confusion in the minds of the majority of high school students about the use of expressions such as X, X^2+3, X+Y, as literal expressions on the one hand and as names for numbers or variables which take numerical values on the other hand. One advantage in using a computer is that these relations can be made very explicit without engaging in philosophical sounding discussions about the names and things named. A good starting point is to recall the program of Lesson 1 in which the instruction TYPE NUMBER caused a particular number, 16, to be typed. Why wasn't the sequence of letters N-U-M-B-E-R typed out? The reason is that the computer understands the word "NUMBER" as the name of 16. Suppose we wished it to type out the word NUMBER rather than the number 16. How could we give it this instruction? In Telcomp an instruction to do this is shown in the following program.

```
SET NUMBER = 4
TYPE "NUMBER"
TYPE NUMBER
```

The first instruction has the effect of making the word NUMBER become the name of the number 4. The second instruction has the effect of typing the word NUMBER. The third instruction has the effect of typing the number 4. Thus, when this program is executed on the computer, it types out the following sequence:

```
NUMBER
4
```

Another example,

```
DEMAND NUMBER
TYPE "THE NUMBER YOU TYPED IS ",NUMBER
```

This program first types NUMBER = and then waits for the user to type in some number. When the user types some number, say 17, the computer types back THE NUMBER YOU TYPED IS 17. The class should be asked to anticipate outputs from some programs of this sort and to write many examples of their own. The point of this lesson can be emphasized by stories like--

Bill: Say your age.
John: Your age.
Bill: No, say what your age is.
John: What your age is.

This kind of game is familiar to children and the appropriate use of quote marks can be emphasized by pointing out how Bill might have avoided the confusion by using quote marks in his demand to John. Thus if Bill says, "Say 'Your age,'" the right answer from John is "Your age," and if Bill says, "Say your age," the correct answer from John is "14" (or whatever his age is).

The corresponding program is:

```
DEMAND AGE
TYPE AGE
TYPE "AGE"
```

After these points are understood, the new instruction READ is explained. READ is the same as DEMAND in so far as it causes the computer to wait for a number to be typed in and gives this number a specified name; however, it does not type anything out. Let's consider the next program.

```
TYPE "TYPE A NUMBER"
READ NUMBER
SET ANSWER = NUMBER*NUMBER
TYPE "THE SQUARE OF YOUR NUMBER IS ",ANSWER
```

This program, when executed, first types TYPE A NUMBER. When the user types some number, say 5, the computer will type back at him THE SQUARE OF YOUR NUMBER IS 25.

General discussion of the lesson should center around this relation of naming. Other terms, such as labeling or pointing to, may be preferred. What should emerge is a sensitivity to the fact that a string of letters can be regarded in two ways --literally, as the thing itself, or as a name, a label, or a pointer, referring to something else, some other thing.

Lesson 3

The programs considered in Lesson 1 use a fixed name for a fixed number during the course of the program. It's true

each time the program runs, each of the names--NUMBER, AREA, RADIUS--refer to a different number; but in the course of running the program at any given time, this number remains fixed. We now consider programs in which some number referred to by a name changes in the course of the program. An analogy in everyday life might be to compare a personal name of an individual man, which remains fixed throughout his life and always refers to the same person, with a name such as "the President of the United States," which may refer to a different person every four years. At the same time we introduce the concept of a loop in a program. The programs we have considered up to now consist of a series of steps executed in the order written, and stopping after the last step is reached. The rule that the machine followed was to carry out the first instruction, then the next, then the next, and so on down the line until it came to the last. If we wish to change this order, we make use of the provision for numbering steps in Telcomp programs.

```
1.1  DEMAND NUMBER
1.2  SET SQUARE = NUMBER*NUMBER
1.3  TYPE SQUARE
1.4  TO STEP 1.1
```

The meaning of the last instruction, TO STEP 1.1, is self-explanatory. The program runs round and round the loop as long as a user keeps on typing values for NUMBER. Each time round, NUMBER refers to a different number, and SQUARE refers to a different number. The columns in the diagram below show the operation of the program.

First Round

	Computer's Action	User's Action	Events Inside the Computer
STEP 1.1 DEMAND NUMBER	Types out: NUMBER =	Types in: 8	NUMBER is now 8.
STEP 1.2 SET SQUARE = NUMBER*NUMBER	(no external	action)	NUMBER*NUMBER i.e., 64 is calculated. SQUARE is now 64.
STEP 1.3 TYPE SQUARE	Types out: 64		(no internal action)
STEP 1.4 TO STEP 1.1	(no external	action)	Causes the computer to go to STEP 1.1.

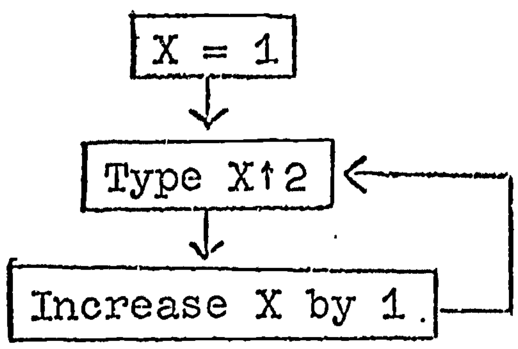
Second Round

STEP 1.1 DEMAND NUMBER	Types out: NUMBER =	Types in: 10	NUMBER is now 10.
STEP 1.2 etc.			

Lesson 4

The last programs lead up to the introduction of the terms variable and value. We'll say that NUMBER, in the previous program, is a variable whose values are successively 8, 10, etc. SQUARE is a variable whose values are 64, 100, etc. We now consolidate the ideas of variable and value by introducing programs in which the program itself changes the values of a variable. The next program will type out the squares of all the numbers for as long as we let the machine run. The program is explained by the following steps or flow diagram.

First step--start with X = 1
Second step--square X and type the result
Third step--increase the value of X by 1
Fourth step--go back to the second step



To write a program to do this, we need a new Telcomp instruction. This is the instruction, replace this by that, represented by X←X+1. The effect of this instruction is to take the old value of X, that is, the X on the right-hand side, add 1 to it, and make this the new value of X. The program we want is as follows:

```
1.1  SET X = 1
1.2  TYPE X
1.3  SET Y = X*X
1.4  TYPE Y
1.5  X←X+1
1.6  TO STEP 1.2
```

The successive passes through this program are shown in the following diagram.

	PASS 1	PASS 2	PASS 3	PASS 4	PASS 5	
X	1	2	3	4	5	etc.
Y	1	4	9	16	25	
X	2	3	4	5	6	

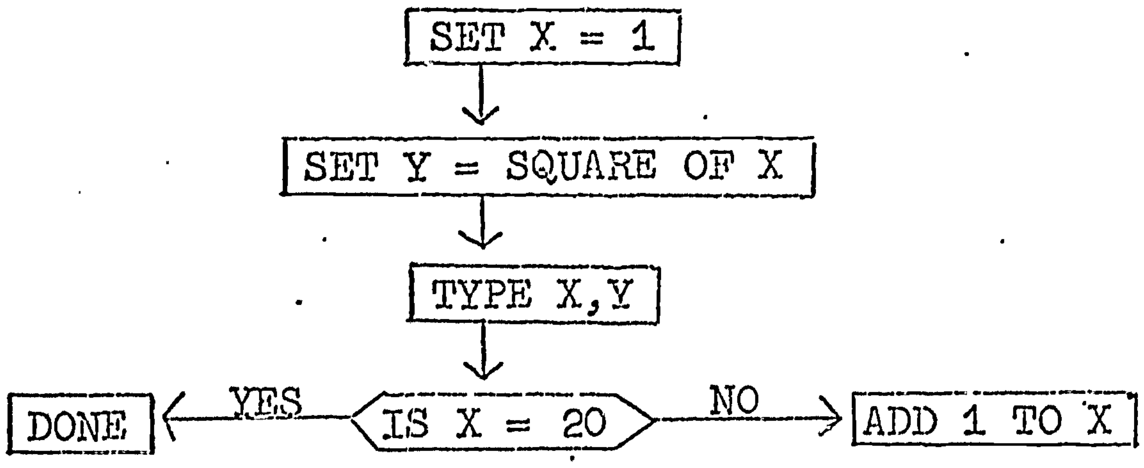
The class might observe and would see on the computer that programs of this sort will go on forever until stopped. Fortunately, we can stop them by pressing an appropriate key on the Teletype. In the following lesson we'll study how to make the program stop itself at whatever stage we wish. First, however, we have the class write a number of programs involving these endless loops.

Lesson 5

This lesson introduces new programming concepts which correspond to the mathematical idea of solving equations or inequalities. Up to now we have considered instructions which are imperatives. We have not been able to make any statement which might be true or false. Telcomp allows us to do so by the use of its conditional command. We introduce this by studying the following program which is the same as the program in the previous lesson except that it will stop when X gets up to the value 20.

```
1.1  SET X = 1
1.2  SET Y = X*X
1.3  TYPE "THE SQUARE OF ",X, " IS ",Y
1.4  DONE IF X = 20
1.5  X←X+1
1.6  TO STEP 1.2
```

While testing this program we introduce branched flow diagrams.



We are now able to write programs which will solve equations by successive trials. We begin by a trivial example. Find a number such that the number plus 3 equals 7.

```
1.1  SET NUMBER = 1
1.2  TYPE NUMBER IF NUMBER + 3 = 7
1.3  NUMBER←NUMBER + 1
1.4  TO STEP 1.2
```

The class might be led up to discovering itself that while this program does find the solution, it contains a small "bug."

After finding the solution, it does not stop. It is a good exercise for a class to discover such bugs and ways to remedy them. When they do, we might teach them another technical feature of Telcomp, namely, the editing facility illustrated by typing the instruction, 1.21 STOP IF NUMBER + 3 = 7. If the condition is satisfied, the computer then will type out STOPPED AT STEP 1.21

When programs dealing with simple problems of this type are understood, we will move to equations whose solutions are less obvious. For some time we will remain in the realm of equations with integral solutions. An example is FIND TWO SUCCESSIVE NUMBERS WHOSE PRODUCT IS 182. The program for this is given a slightly different form to encourage flexibility in thinking about the sequencing of program steps.

```
1.1  SET X = Ø
1.2  X←X+1
1.3  TO STEP 1.2 IF X*(X+1) < > 182
1.4  TYPE X
```

Programs on inequalities have exactly the same form. For example, to type out all whole numbers, x, (1 to 100) for which $10x - x^2 > 18$:

```
1.1  SET X = Ø
1.2  X←X+1
1.3  TYPE X IF 1Ø*X - X*X > 18
1.4  DONE IF X > 1ØØ
1.5  TO STEP 1.2
```

General discussion on this lesson could emphasize the difference between:

imperatives like SET X = 1 which are neither true nor false,

statements like X > 2 which are true for some values and false for others,

identities like X + 1 = 1 + X which are always true,

and contradictions like X + 1 = X which are never true.

Consolidating exercises could range from questions such as--

Are the following pairs of instructions equivalent:

TYPE X IF X > 2	TYPE X
TYPE X IF X > 2	TYPE X IF X * X > 4
TYPE X IF X = X	TYPE X
etc.	

to writing a program to the following specifications--

The program should type "THIS IS A PROGRAM TO SOLVE EQUATIONS OF THE FORM X + A = B . TYPE THE VALUES OF A AND B." If, for example, the user gives A and B the values 3 and 7, respectively, the program should type "THE SOLUTION OF X + 3 = 7 IS X = 4."

Lesson 6

The lesson is designed to illustrate another example of how formalizing a conceptually simple procedure might motivate the introduction of a mathematical concept which might otherwise seem somewhat mysterious. The problem for students is to write a program to the specifications:

The user will type in two lists of numbers. The program will type out all numbers which appear on both lists.

The first subproblem is to decide how the computer can name the members of a list of numbers. If we know when we write the program how many numbers there are to be, we could call them FIRST, SECOND, THIRD, and so on. The following program does this for lists of three numbers.

```
1.1  TYPE "TYPE FIRST LIST"
1.2  READ FIRST
1.3  READ SECOND
1.4  READ THIRD
1.5  TYPE "TYPE SECOND LIST"
1.6  READ NUMBER
1.7  TYPE NUMBER IF NUMBER = FIRST
1.8  TYPE NUMBER IF NUMBER = SECOND
1.9  TYPE NUMBER IF NUMBER = THIRD
1.10 TO STEP 1.6
```

If, however, we do not know how many numbers there are to be, we need a more flexible method of naming. The introduction of "subscript" or "functional" notation covers this need and introduces a valuable mathematical idea in a concrete form. Telcomp allows us to use x[1], x[2], x[3],.... as names and x[NAME] (where NAME is any number name) as a general form. The following program shows how this is used. This program simply reads a list and types it back.

```
1.1  TYPE "HOW MANY NUMBERS IN YOUR LIST?"
1.2  READ N
1.3  SET FOO = 1
1.4  TYPE "TYPE NUMBER"
1.5  READ X[FOO]
1.6  TO PART 2 IF FOO = N
1.7  FOO ← FOO + 1
1.8  TO STEP 1.4
2.0  SET FOO = 1
2.1  TYPE X[FOO]
2.2  DONE IF FOO = N
2.3  FOO ← FOO + 1
2.4  TO STEP 2.1
```

When this is understood, we show how to simplify it using the Telcomp FOR statement. The program becomes:

```
1.1  TYPE "HOW MANY NUMBERS ON YOUR LIST?"
1.2  READ N
1.3  READ X[FOO] FOR FOO = 1:1:N
1.4  TYPE X[FOO] FOR FOO = 1:1:N
```

We now turn back to the original problem. We first write programs to read in the two lists where NUMBER will be called FIRST[1], FIRST[2],... and SECOND[1], SECOND[2],....

```
1.1  TYPE "HOW MANY NUMBERS ON FIRST LIST?"
1.2  READ FIRSTN
1.3  TYPE "PLEASE TYPE YOUR FIRST LIST"
1.4  READ FIRST[FOO] FOR FOO = 1:1:FIRSTN
1.5  TYPE "HOW MANY NUMBERS ON SECOND LIST?"
1.6  READ SECN
1.7  TYPE "PLEASE TYPE YOUR SECOND LIST"
1.8  READ SECOND[FOO] FOR FOO = 1:1:SECN
```

The teacher should carefully discuss the fact that FOO can be used twice and that we could have written:

```
1.6  READ SECOND[BOO] FOR BOO = 1:1:SECN
```

with exactly the same effect. This is valuable in itself and as a preparation for STEP 1.7 which will have to be carefully presented with trace diagrams etc.

```
1.7  TYPE FIRST[I] IF FIRST[I] = SECOND[J] FOR I = 1:1:FIRSTN
                                           FOR J = 1:1:SECN
```

If the double FOR statement is too difficult, we might lead up to it by writing the same program first with two single FOR statements. When these programs are thoroughly understood, we rewrite the equation-solving programs using FOR statements and introduce essentially similar problems on simultaneous equations and simultaneous inequalities.

General discussion should now consolidate the concept of variable and of equations as placing constraints on possible values.

Lesson 7

Up to now we have dealt only with equations where solutions are whole numbers. The study of search procedures for rational and irrational numbers might lead to valuable insights into concepts of the number system.

We begin this topic by recalling the search program for equations and asking whether it would succeed for:

$$x + 21 = 69$$
$$3x + 4 = 7$$
$$3x + 2 = 7$$

The class should understand that it would fail for the third of these, and why. We then construct a fraction searching program which goes from 1 to 100, for example, first in steps of units, then in steps of 1/2, then in steps of 1/3, etc. To do this, instead of replacing X by X+1, X←X+1, we replace X by X+FRACTION where FRACTION is 1/1 on the first round, 1/2 on the second, and so on. In other words, FRACTION is always 1/DENOMINATOR and DENOMINATOR is 1, 2, 3, and so on.

We do not have enough experience to know whether the following program is too difficult at this point, but with suitable backing discussion, it should be presentable.

```
1.1  SET DENOMINATOR = 1
1.2  SET FRACTION = 1/DENOMINATOR
1.3  SET X = Ø
1.4  X←X+FRACTION
1.5  TO STEP 2.1 IF 3*X+2 = 7
1.6  TO STEP 3.1 IF X = 1ØØ
1.7  TO STEP 1.4

2.1  TYPE X

3.1  DENOMINATOR ← DENOMINATOR + 1
3.2  TO STEP 1.2
```

7.3 Summary of Findings

1. Throughout the high schools the regular classroom work in mathematics and science was often augmented by independent student projects utilizing the computer. The high school teachers participating in the project summarized these activities during the 1965-1966 school year as follows: "As the year progressed ... ideas which were generated in a classroom discussion and elaborated by a student at the terminal in his spare time produced some very original and sophisticated work in diverse fields. Programs were written on topics from geometry, calculus, probability, number theory, game theory, English, Latin, music, photography, cryptography, and psychology. Many of these students ... in order to achieve desired results were forced to generalize and abstract in a manner which, under ordinary circumstances, they never would have done. Theoretical concepts were made concrete, seemingly divergent ideas were connected, a logical approach to problem-solving and an analytical way of thinking developed, and a cooperative interaction between students was observed. ..." Most students involved in these extracurricular projects were highly motivated and mathematically skilled. In some instances, however, the use of the computer stimulated students of average ability and unexceptional initiative to work enthusiastically and effectively on their own projects.

2. The use of Telcomp as a mathematics laboratory to supplement the conventional classroom presentation was most effective in sixth, eleventh, and twelfth grade courses, somewhat less in ninth grade algebra. Although this mode of use was generally hampered in several ways (these are discussed in the following section), it was established as a new and valuable way of improving the curriculum. Though individual explorations in each of the schools differed in many respects, each had profound effects of the same sort on large numbers of students -- most students found algorithmic work on the computer an intensely involving and rewarding experience.

3. Work done during the first year of the project established that the computer could be effectively used as a mathematics laboratory to make possible an experimental approach to problem solving. The evident educational benefits -- conceptual and notivational -- which could come from this extra dimension of mathematical experience were clearly seen from both extracurricular and curriculum-supplementary explorations. We hypothesized that these benefits could be realized in a mathematics curriculum utilizing the computer to provide an algorithmic framework for the presentation of mathematics. In order to test this hypothesis in a preliminary way, during the limited time remaining in the project, we developed the sequence of lesson plans for the beginning part of a laboratory course in algebra (reproduced in

Section 7.2.1). The lessons were used in one of the participating schools in the early weeks of a 9th grade introductory algebra course. Although the sequence is informal and preliminary and the teacher was not specially trained in teaching with it, he had extraordinary success. His pupils experienced little difficulty in comprehension of, or manipulation with, the algebra of variables, expressions, and equations. In his teaching experience, of several years, comparable students of algebra had never fared so well

4. Project H-212 has had direct impact and influence in other educational developments and research. Some of the ramifications of the project are listed in Appendix 4.

7.4 Problems

Several factors constrained the scope and depth of the research.

1. Resource Limitation

Instead of a single computer terminal at each of the schools, there probably should have been several terminals in a single school. In each of the high schools from 100 - 300 students tried desperately to get regular use of the computer at least twice a week and this was impossible to schedule within the time available. Even when the schools opened for such use during evenings and weekends, it was difficult to provide adequate time for all active students on a regularly scheduled basis.

2. Supplementary Course Materials

The summer phase was not of sufficient duration to permit the preparation of extensive lesson plans and associated teaching materials. Thus teachers did not begin with elaborated teaching plans, except in a few cases.

3. Teacher Training

Most teachers had no previous classroom experience in teaching programming or the use of programming in teaching mathematics. The summer teacher training program was sufficient to bring most teachers to some point of fluency in programming but not sufficient to enable many to proceed confidently to incorporate programming smoothly into mathematics teaching.

4. Classroom Demonstrations

The computer could not generally be used in the classroom for live teacher or student demonstrations of mathematical procedures, either because schools did not have a closed-circuit television system or could not conveniently use television systems in the classroom. Thus, the need for an inexpensive projection system for use with a computer terminal was recognized. The subcontractor providing technical support to the project, Bolt Beranek and Newman, Inc., subsequently designed and developed such a device.

8. Conclusions and Recommendations

We state our principal conclusions in summary form.

(1) It is possible to construct programming languages of great expressive power yet so simple to learn that they can be effectively taught to elementary school children.

(2) Children are easily motivated to write programs at computer consoles. This kind of mathematical activity is immensely enjoyable to children generally, including those not in the top levels of mathematical ability.

(3) Programming work facilitates the acquisition of rigorous thinking and expression. Children impose the need for precision on themselves through attempting to make the computer understand and perform their algorithms.

(4) A series of key mathematical concepts such as variable, equation, function, and algorithm, can be presented with exceptional clarity in the context of programming.

(5) The use of a programming language effectively provides a working vocabulary, an experimental approach, and a set of experiences for discussing mathematics. Mathematical discussion among high school students, relatively rare in the conventional classroom, was commonplace in this laboratory setting.

(6) Computers and programming languages can be readily used in either of two ways in the mathematics classroom,

- (a) By individual students for independent study on extracurricular problems or special projects.
- (b) As a laboratory facility to supplement regular classroom lecture and discussion work. In this mode students are given assigned problems to work out at the computer.

(7) A third way of using computers and programming, that might have radical implications for the presentation of mathematics, was uncovered - the concept of teaching programming languages as a conceptual and operational framework for the teaching of mathematics.

We have a single recommendation for further research. The development of student and teaching materials for using computers and programming to supplement regular teaching has already begun through projects stimulated by H-212. The, potentially even more important, effort to develop an integrated mathematics laboratory curriculum should now be undertaken. The object would be to show (1) that programming can be used as a conceptual framework for the presentation of mathematics and (2) that such a presentation could make a radical improvement in mathematics education.

APPENDIXES

APPENDIX I

Section A. History and Sequence of Project H-212

Section B. Original Evaluation Design, Later Discarded

APPENDIX I

Section A: History and Sequence of Project H-212

During the past several years, teachers and administrators have evinced a rapidly growing interest in the use of the electronic computer and its associated equipment as a means of providing more effective teaching in the classroom and laboratory. Teachers who have attended National Science Foundation (NSF) institutes and programs have been introduced to the effectiveness of the computer as a powerful tool for solving problems, many of which are far too complex to include in the usual science and mathematics curriculum because of the excessive length of time required for solution by the usual methods.

A growing number of teachers have had the opportunity to work closely with industrial and scientific organizations where computer facilities were in use, have developed skills in the use of the equipment, and have in turn made use of this knowledge to complement and improve their own classroom teaching techniques. This has been particularly true in the vicinity of the larger metropolitan areas or in locales where concentrations of scientific, research, and development activities exist.

In recognition of this interest, the Massachusetts Department of Education has, since 1961, explored the use of the computer as an instructional aid for the mathematics classroom and laboratory. It has worked closely with the computer industry, the New England School Science Advisory Council (NESSAC), an independent agency representing more than forty professional, scientific, and technical societies, and the Association for Computing Machinery (ACM) in establishing pilot programs in computer-oriented mathematics instruction in selected local school systems in the Commonwealth.

In 1962 the Department of Education sponsored a 16 week course in Computer Mathematics for high school science and mathematics teachers, in cooperation with the Electronics Data Processing Division of the Minneapolis-Honeywell Company in Wellesley Hills, Massachusetts. Twenty-five teachers completed the program, and were familiarized with the basic concepts of computers and computer programming.

The demand for this course so far exceeded the number that could be accommodated that a similar course was presented during the summer of 1963, this time with the assistance of

the International Business Machines Corporation. Twenty teachers attended this summer session, using an IBM 1620 computer and associated equipment.

Pilot High School Program

Concurrently with the initiation of the teacher programs just described, the Department of Education requested the assistance of the Association for Computing Machinery (ACM), Greater Boston Chapter, in establishing a pilot course in computer mathematics and programming for high school students. In February 1963, such a course was begun at Westwood High School, Westwood, Massachusetts, under the joint sponsorship of the Westwood School Committee, ACM, and the State Department of Education. Staff computer scientists at Bolt Beranek and Newman Inc. of Cambridge, Massachusetts, served as instructors for the program.

The class was composed of 20 mathematically able students of the ninth, tenth, and eleventh grades, selected by the head of the Mathematics Department at Westwood High School. The class met after school for one hour and a half, one afternoon a week. Equipment used in the course was furnished by Bolt Beranek and Newman Inc., and by Digital Equipment Corporation (DEC) of Maynard, Massachusetts. The PDP-1 computer of Bolt Beranek and Newman Inc. was used, and many of the students successfully completed and debugged at least one program. Of the three grade levels of students included in the course, the ninth graders showed the most substantial accomplishment. Some students developed strong interest in programming and wrote fairly sophisticated programs, while others became interested in logical design and planned their own computer systems.

State-wide Seminar Program

The interest of local school systems in establishing instruction in computers and computer-oriented mathematics grew sharply during the 1963-1964 academic year. A state-wide program, Seminars and Advanced Studies for Able Students of Science and Mathematics, was established in the fall of 1963. Selected schools served as regional centers for instruction in advanced areas of science and mathematics. Twenty-eight such instructional centers were established, and more than 1500 students attended these extracurricular sessions, entirely on a voluntary basis. As an indication of student interest in computers, it is significant to note that twelve of the twenty-eight seminar groups requested and received instruction in computers and computer-oriented mathematics. The instructors for the program, all of whom donated their time, were drawn from the membership of NESSAC, the Association for Computing Machinery, and scientific and technical organizations.

In general, these pilot programs in computer-oriented mathematics instruction taught students basic concepts of computer operation, capabilities, and programming. When actual experience with a computer was needed, students were taken on a field trip to a computer facility, usually on a Saturday, so as to prevent interference with the daily school program. However, this provided at best only limited access to the computer itself.

The results of these pilot instructional programs were quite satisfactory, proving to be both feasible and rewarding to the student, but they did not give him the actual "hands-on" experience with the system necessary to provide familiarity with new methods of computation. It was concluded, therefore, that if it were possible for a mathematics teacher to have a large-scale computer at his disposition in the classroom at times appropriate to his class schedule, he and his students could experiment readily with sophisticated mathematical concepts as an extension of the usual classroom and laboratory instruction. However, such a facility was beyond the reach of the usual public school system's budget.

Curriculum Improvement Project H-212

At this time certain significant developments in computer technology seemed to demonstrate unprecedented potential for improvement of the educational process. These major advances in computer science were:

(1) the establishment of large multi-user, multi-purpose computer "time-sharing" systems, and

(2) the introduction of techniques that permit direct and sustained interactions between a person and a computer-based information processing system.

With the firm conviction that the application of this new technology to the improvement of instruction in mathematics would provide results of benefit to the entire educational community, the Massachusetts Department of Education filed a research proposal with the Cooperative Research Branch, United States Office of Education. A major grant of $176,000 was received from this agency in June 1965, to carry out this research as Project H-212, entitled "Teaching Mathematics Through the Use of a Time-Shared Computer."

The procedures and time schedule followed by the project were modified from those established in the original planning. The following section of the original project proposal, as approved by the Bureau of Research, is included herein as a reference.

"This project will be conducted in three phases:

1. An in-school pilot phase conducted during the spring of 1965 to install and test equipment, to familiarize instructional personnel with the use of the equipment, to carry out limited teaching experiments, to establish tentative instructional format, and to experiment with the evaluation procedures.

2. A summer study session involving educators and members of the advisory board (see below), to report on the results of the pilot phase, to provide training for additional teachers involved in the program, to review and improve the curriculum materials for the third phase.

3. The main experimental phase of the project conducted for a full school year at the three grade levels described above.

B. Summer Study Session

We propose to conduct a work session during the summer of 1965 utilizing the facilities at Bolt Beranek and Newman Inc.

A group of 30 educators involved in the teaching of mathematics at the elementary and secondary school levels, scientists involved in the time-shared use of computers, and members of the project advisory board will be assembled for a 6-week session. This work session will aim to:

1. teach the mathematics teachers the fundamentals of computer usage and the potentials of a real-time computer in the mathematics and science classroom;

2. review and improve the curriculum units developed for the classes using the computer;

3. evaluate the results of the initial phase of the experiment conducted during the spring semester of 1965;

4. establish a teacher's user-group to write and share programs over the course of the study;

5. thoroughly test the computer programs developed during the initial phase of the experiment;

6. develop new computer programs to be used during the school year 1965-1966."

Unexpected delays in obtaining the necessary compliances with regulations and the subsequent negotiation of the contract for the project produced a delay of several months in the schedules originally proposed. Project activities began on June 1, 1965, five months later than had been planned. Because of this late start, the in-school pilot phase scheduled for the spring of 1965 could not be conducted. The familiarization of teachers with the use of the computer terminals, the limited teaching experiments to establish tentative format, and experimentation with the procedures for evaluation were not able to be carried out.

It was evident that a major change in the schedule of project activities and procedures was necessary if the major objectives were to be adequately met. The situation was reviewed by the project director and the project monitor appointed by the Office of Education. The following changes were made in the project procedures:

1. The summer session, scheduled for 6 weeks in July and August 1965, would carry out the activities originally scheduled for the in-school pilot phase for spring 1965.

2. The activities originally planned for the summer session would be carried out in the schools during the first semester of the 1965-1966 school year. The development of curriculum materials and their evaluation would be done in the second semester of the school year.

3. Since a full school year was not now available for the main experimental phase of the project, it was not possible to comply with the tightly controlled evaluation plan that had been originally established. In order to provide the maximum opportunity for the project to achieve its objectives, the evaluation plan was discarded as being no longer applicable.

4. The basic structure of the project was changed from its original plan to that of a methodological study. A number of relatively autonomous explorations would be conducted at the participating schools to learn how students could work most effectively with the computer as a teaching and learning aid. The results of these explorations would then be combined and analyzed to get a judgement as to the applicability of the computer for the improvement of mathematics instruction.

There would be no change in the grade levels originally specified for the participants. However, an additional sixth grade school was added, and additional funding was provided through amendment of the contract for the project for the added computer services required for them to participate.

5. A substantial number of computer programs were developed by the participating schools during the second semester of the 1965-1966 school year. In order to evaluate them and to begin documentation of the project activities, a group of teachers and students from the participating schools worked with the project staff for six weeks during July and August 1966. The materials produced by this group have been included in the final report for Project H-212.

6. It became apparent in early September 1966, that a substantial amount of work remained to be done in screening the instructional materials developed by the participating schools. Although this task was undertaken by the summer group of teachers and students, several hundred of a total of more than five hundred computer programs remained to be processed for documentation. It was felt that this could be accomplished most effectively by asking the participating teachers to do this in their own schools. However, the existing contract for Project H-212 was written to terminate on October 31, 1966, and did not allow time to complete this work.

   An examination of the basic contract for the project indicated that the expenditures through August 31, 1966, were at a rate of approximately 80% of the total grant. As a result, enough funds would remain expendable as of October 31st to continue the research program for three additional months through January 31, 1967. This extension would be at a level of activity only slightly less than that of the normal program involving all of the participating schools.

   Accordingly, a request for extension of the termination date of contract OE 5-10-320 for Project H-212 from October 31, 1966 to January 31, 1967 was filed with the Bureau of Research in September. This request was approved, and the new termination date became effective. A second contract extension was later approved; the final termination of project activities was on April 30, 1967.

APPENDIX I

Section B. Original Evaluation Design, Later Discarded.

Evaluation Procedures

The value of this experiment cannot be measured exclusively by statistical gains on one or more psychometric instruments; such evaluation can, however, be used for exploring the types of impact resulting from the program, as for example:

1. increased student knowledge about mathematics;
2. increased student knowledge about digital computers;
3. changed student attitudes towards the role of the computer in our society;
4. opinions, either positive or negative, among teachers, school administrators, and parents, about the value, and advisability of a computer-oriented mathematics program.

Quantitative information about student change can tell us more than whether or not the program is achieving its goals; it can give us information (by means of item analyses, etc.) on how to modify procedures to strengthen various aspects of the program.

Basically, the evaluation model of the experiment is a two-way design with covariance adjustment for premeasures. We assume assignment to treatment is random except for the measures controlled.

		TREATMENT		
Teacher	Grade	Time-Sharing Class	Non-Time-Sharing Class	Control
A	11	$Class_{11}$	$Class_{12}$	$Class_{13}$
B	9	$Class_{21}$	$Class_{22}$	$Class_{23}$
C	6	$Class_{31}$	$Class_{32}$	$Class_{33}$

The following criterion measures to be applied to all classes appear most suitable:

1. a mathematics achievement test for the computer-oriented curriculum (to be developed).
2. Sequential Test of Education Progress (STEP), mathematics form 3A for grades 6, and 9; form 2A for grade 11.

These will be administered as postmeasures to all groups.

As premeasures used to control the criteria for initial differences among groups, we have;

1. Otis Test of Mental Ability;
2. a pre-test for the computer-oriented curriculum;
3. STEP mathematics forms 3B and 2B;
4. prior mathematics grade.

Since the STEP forms are not comparable from grades 6 and 9 to grade 11, the analyses of this criterion will be split, yielding one two-way analysis (grades 6 and 9 vs. treatment) and a one-way analysis of grade 11 over the three treatments.

The analysis of the first criterion may follow the 3 x 3 model, provided the test can be made general enough to cover the grade spread involved. This is feasible, since the students for the most part will come with little prior knowledge of computers.

From these three covariance tables we can test the following hypotheses when the students' criterion measures are corrected for the premeasures:

1. treatment has no effect on either criterion;
2. grade level has no effect (6 and 9 only for STEP);
3. the effect of treatment does not depend on the grade in which it is given (as above for STEP).

Further analysis will be done if these hypotheses are rejected with sufficient confidence. This would involve a detailed diagnosis of the pre-post changes on individual items, using McNemar's test. Such pre-post item analysis will also be done on the student attitude questionnaire data.

It may be noted that in the above design the effect of grade is confounded with behavioral differences among teachers. To investigate the effect of individual teachers on the various treatments, the procedure will be repeated with two additional sets of ninth grade classes. To analyze this effect, we will use the following model representation for a two-way analysis of covariance. This analysis will enable us to observe the teacher effect, the treatment effect, and the interaction between teacher and treatment within the ninth grade.

		TREATMENT		
Teacher	Grade	Time-Sharing Class	Non-Time-Sharing Class	Control
B	9	$Class_{21}$	$Class_{22}$	$Class_{23}$
D	9	$Class_{41}$	$Class_{42}$	$Class_{43}$
E	9	$Class_{51}$	$Class_{52}$	$Class_{53}$

The Relation of Student Involvement to Various Criteria

Assuming we find that our treatment affects one or more of the criteria, we may then look at the relation between the degree of student involvement during the course and student standing. The time-shared computer provides a means for keeping a detailed cumulative record of the students' performance and behavior. For example, records may quite easily be kept as to the amount of time the student uses the computer and the frequency variation of his errors as he progresses.

To observe how the amount of time the student puts in affects his criterion measure, we may wish to look at several partial correlations, such as:

1. the relation between time used and post-test results, when IQ and pre-test results are held constant;
2. the relation between prior attitude towards the program and computer time used, when controlling for IQ and pre-test.

Many such relations will be found interesting and important. For example, the first relation may tell us something about the importance of "exposure;" the second may suggest whether or not pre-disposition is related to the degree to which the student applies himself to the study. In addition, we might wish to gather information on whether the intervention of this course adversely affects students' grades in other courses.

APPENDIX II

Assigned Problems and Demonstrations Used to Supplement Instruction

Section A. Sixth Grade Programs

Section B. Ninth and Tenth Grade Programs

Section C. Eleventh and Twelfth Grade Programs

APPENDIX II

Assigned Problems and Demonstrations Used to Supplement Instruction

SECTION A: Sixth Grade Programs

1. RECIP - Use of Reciprocals
2. LCM - Least Common Multiple
3. TRISUM - Product of Three Consecutive Numbers
4. LOWPRI - Prime Number Generation
5. BOX - Volume and Surface Area of a Box
6. FAC - Integral Divisors of a Number
7. NUMGES - Guessing a Number
8. FRAC - Manipulation of Fractions
9. PRM - Identifying Prime Numbers

RECIP --A Demonstration Program to Suggest the Use of Reciprocals for Simplifying Division of Integers by Rationals.

This program was written by Barbara J. Capron, a teacher in the Kendall School, Belmont, for use in her sixth grade class. It was designed to lead students to observe (and, presumably, to prove necessary) a general pattern recurring in a set of individual calculations. The same device has been used with TELCOMP elsewhere to suggest the plausibility of possible trigonometric identities.

Listing of Program:

```
1.1 SAY TWO FACTORS WHOSE PRODUCT IS 1 ARE CALLED RECIPROCALS
1.2 SAY 3/4 IS THE RECIPROCAL OF 4/3
1.3 SAY 4/3 IS THE RECIPROCAL OF 3/4
1.31 LINE
1.4 SAY WE USE RECIPROCALS WHEN WE DIVIDE BY RATIONAL NUMBERS
1.41 TYPE 3/(1/4),3*4
1.42 TYPE 8/(2/5),8*(5/2)
1.43 TYPE 10/(5/6),10*(6/5)
1.44 TYPE 7/10/(1/100),7/10*(100/1)
1.45 TYPE 2.5/(2/5),2.5*(5/2)
1.5 SET YES=0
1.51 SET NO=1
1.52 SAY A/B IS THE RECIPROCAL OF B/A
1.53 SAY WE WILL DIVIDE ANY NUMBER N BY ANY RATIONAL NUMBER A/B
1.6 DEMAND N
1.61 DEMAND A
1.62 DEMAND B
1.63 TYPE A/B,B/A
1.7 TYPE N/(A/B),N*(B/A)
1.8 SAY SHALL WE DO ANOTHER DIVISION?
1.81 DEMAND ANS
1.82 TO STEP 1.6 IF ANS=YES
1.9 SAY WHAT DO YOU OBSERVE?
1.91 SAY HOW CAN YOU USE RECIPROCALS FOR DIVIDING RATIONAL NUMBERS?
```

Operation of Program:

```
TWO FACTORS WHOSE PRODUCT IS 1 ARE CALLED RECIPROCALS
3/4 IS THE RECIPROCAL OF 4/3
4/3 IS THE RECIPROCAL OF 3/4

WE USE RECIPROCALS WHEN WE DIVIDE BY RATIONAL NUMBERS
     3/(1/4) =      12
         3*4 =      12

     8/(2/5) =      20
     8*(5/2) =      20

    10/(5/6) =      12
    10*(6/5) =      12

7/10/(1/100) =      70
7/10*(100/1) =      70

   2.5/(2/5) =       6.25
   2.5*(5/2) =       6.25

A/B IS THE RECIPROCAL OF B/A
WE WILL DIVIDE ANY NUMBER N BY ANY RATIONAL NUMBER A/B
N=1/2
A=1
B=16
          A/B =        .0625
          B/A =      16

      N/(A/B) =        8
      N*(B/A) =        8

SHALL WE DO ANOTHER DIVISION?
ANS=YES
N=99
A=12
B=15
          A/B =        .8
          B/A =      1.25

      N/(A/B) =    123.75
      N*(B/A) =    123.75

SHALL WE DO ANOTHER DIVISION?
ANS=NO
WHAT DO YOU OBSERVE?
HOW CAN YOU USE RECIPROCALS FOR DIVIDING RATIONAL NUMBERS?
←
```

LCM - A Program to Calculate Least Common Multiple.

Billy Davenport, a sixth grade student at Payson Park School, Belmont, wrote this program. The problem is not trivial for a sixth-grader. In this instance the solution exhibits understanding and ingenuity. It illustrates the comprehension of mathematical algorithms attained by some of the sixth-graders given even the little opportunity afforded during the year for individual student use of the TELCOMP terminal in the elementary schools.

Listing of Program:

```
1.1 LET LCM=A*I
1.2 TO STEP 1.5 IF FP(LCM/B)=0
1.3 LET I=I+1
1.4 TO STEP 1.1
1.5 TO STEP 1.7 IF FP(LCM/C)=0
1.6 TO STEP 1.3
1.7 TYPE LCM


2.1 DEMAND A
2.2 DEMAND B
2.3 DEMAND C
2.4 LET I=1
2.5 DO PART 1
2.6 TO STEP 2.1
```

Operation of Program:

```
 DO PART 2
A=8
B=16
C=4
            LCM =     16

A=17
B=19
C=2
            LCM =    646

A=10
B=20
C=30
            LCM =     60

A=2
B=3
C=5
            LCM =     30

A=99
B=99
C=99
            LCM =     99
```

TRISUM - A Program to Find the Product of Three Consecutive Even (or Three Consecutive Odd) Numbers Whose Sum is Given.

This nice program was written by Peter Bass of the sixth grade at Payson Park School. Mathematical problems of this type are interesting and instructive to elementary students. The basic problem here, of course, is to find the three summands. The product calculation is then trivial. (True, the requirement for calculating the product may fool some students.) Mr. Bass has it that, if the sum is X then the summands must be X/3, X/3 + 2, and X/3 - 2. How did he figure that out? Note that his solution is not limited to integer summands -- when is a repeated decimal even or odd? Problems like this, along with the associated programs, give rise to real mathematical issues for class discussion.

Listing of Program:

```
1.1 DEMAND X
1.2 LET M=X/3
1.3 LET A=M+2
1.4 LET B=M-2
1.5 TYPE B,M,A
1.6 LET P=B*M*A
1.7 TYPE P
1.8 TO STEP 1.1
```

Operation of Program:

```
X=30
            B =     8
            M =    10
            A =    12

            P =   960

X=654
            B =   216
            M =   218
            A =   220

            P =     1.035936*10↑7

X=349
            B =   114.3333
            M =   116.3333
            A =   118.3333

            P =     1.573925*10↑6

X=849
            B =   281
            M =   283
            A =   285

            P =     2.266405*10↑7
```

LOWPRI - A Program to Generate the Prime Numbers Less Than 100.

James Ktona, the author, is another student at Payson Park School. He has written a straightforward and short program for calculating the low primes. It is an impressive piece of work. Can you write a shorter program? The teacher, Edith Bixby, states that this program is a favorite one of the children in the class. Perhaps they understand why it works. Can you write a program, utilizing the same principle, to calculate the primes less than N, where N is not known in advance?

Listing of Program:

```
1.1 TO STEP 1.6 IF FP(PRI/2)=Ø
1.2 TO STEP 1.6 IF FP(PRI/3)=Ø
1.3 TO STEP 1.6 IF FP(PRI/5)=Ø
1.4 TO STEP 1.6 IF FP(PRI/7)=Ø
1.5 TYPE PRI
1.6 DONE

2.1 DO PART 1 FOR PRI=8(1)1ØØ
```

Operation of Program:

```
←DO PART 2

        PRI =   11
        PRI =   13
        PRI =   17
        PRI =   19
        PRI =   23
        PRI =   29
        PRI =   31
        PRI =   37
        PRI =   41
        PRI =   43
        PRI =   47
        PRI =   53
        PRI =   59
        PRI =   61
        PRI =   67
        PRI =   71
        PRI =   73
        PRI =   79
        PRI =   83
        PRI =   89
        PRI =   97
```

BOX - Surface Area and Volume

This program calculates the volume and surface area of a rectangular box, given its length, width, and height. These values are entered by the student.

Listing of Program:

```
1.1 DEMAND L
1.2 DEMAND W
1.3 DEMAND H
1.4 SET V=L*W*H
1.5 SET SA=(L*W*2)+(W*H*2)+(H*L*2)
1.6 TYPE V,SA
```

Operation of Program:

```
+DO PART 1 FOR A =1:1:11

          L=4
          W=4
          H=4

          V=     64
          SA=    96


          L=6
          W=6
          H=2

          V=     72
          SA=    120


          L=8
          W=2
          H=4

          V=     64
          SA=    112


          L=4
          W=4
          H=2

          V=     32
          SA=
          64
```

L=1
W=2
H=3

V= 6
SA= 22

L=2
W=2
H=3

V= 12
SA= 32

L=4
W=2
H=3

V= 24
SA= 52

L=4
W=2
H=6

V= 48
SA= 88

L=5
W=2
H=4

V= 4Ø
SA= 76

FAC - Factors of a Number

This program generates all of the integral divisors of a number selected by the student.

Listing of Program:

```
1.1 DEMAND N
1.2 SET CNT =1
1.3 SET FAC =N/CNT
1.4 TO STEP 1.8 IF FP (FAC)=Ø
1.5 SET CNT =CNT+1
1.6 TO STEP 1.1 IF CNT > N
1.7 TO STEP 1.3
1.8 TYPE FAC
1.9 TO STEP 1.5
```

Operation of Program:

```
←DO PART 1 FOR X = 1:1:15
           N=144
          FAC=  144
          FAC=   72
          FAC=   48
          FAC=   36
          FAC=   24
          FAC=   18
          FAC=   16
          FAC=   12
          FAC=    9
          FAC=    8
          FAC=    6
          FAC=    4
          FAC=    3
          FAC=    2
```

NUMGES - Number Guessing

This program uses the strategy of binary search to guess a number that the student is thinking of, using the smallest possible number of tries.

Listing of Program:

```
1.1 TYPE "GUESS A NUMBER BETWEEN 1 AND 1ØØ."
1.2 HI=1.111, LOW=1.ØØ1, RIT=1.ØØ2
1.21 ANS=1.ØØ3
1.3 T=1ØØ, 8=Ø
1.31 TYPE "I GUESS 1ØØ.  AM I HI OR RIT?"
1.31Ø2 DEMAND ANS
1.4 G=IP((T-B)/2)+8
1.51 TO STEP 1.9 IF ANS=RIT
1.52 PRINT "I GUESS ",G," .  AM I HI, LOW, OR RIT?",#
1.6 DEMAND ANS
1.7 TO STEP 1.91 IF ANS=HI
1.8 TO STEP 1.93 IF ANS=LOW
1.9 TYPE "WANT TO GUESS AGAIN?" IF ANS=RIT
1.9Ø1 DONE IF ANS=RIT
1.91 T=G
1.92 TO STEP 1.4
1.93 B=G
1.94 TO STEP 1.4
```

Operation of Program:

```
GUESS A NUMBER BETWEEN 1 AND 1ØØ.
 I GUESS 1ØØ.  AM I HI OR RIT?
        ANS=HI
I GUESS 5Ø .  AM I HI, LOW, OR RIT?
        ANS=LOW
I GUESS 75 .  AM I HI, LOW, OR RIT?
        ANS=LOW
I GUESS 87 .  AM I HI, LOW, OR RIT?
        ANS=LOW
I GUESS 93 .  AM I HI, LOW, OR RIT?
        ANS=LOW
I GUESS 96 .  AM I HI, LOW, OR RIT?
        ANS=LOW
I GUESS 98 .  AM I HI, LOW, OR RIT?
        ANS=HI
I GUESS 97 .  AM I HI, LOW, OR RIT?
        ANS=RIT
WANT TO GUESS AGAIN?
```

FRAC - Fraction Manipulator

This program was used at Winn Brook School, Belmont. It adds any two fractions A/B and C/D and computes the sum as an integral part and a fractional part in reduced form.

Listing of Program:

```
1.Ø1 DEMAND A,B,C,E
1.Ø2 I=1
1.Ø3 D=B*I
1.Ø4 I=I+1
1.Ø5 TO STEP 1.Ø7 IF FP(D/E)=Ø
1.Ø6 TO STEP 1.Ø3
1.Ø7 N=((D/B)*A)+((D/E)*C)
1.Ø8 TO STEP 1.2Ø IF N>D
1.Ø9 J=N
1.1Ø TO STEP 1.16 IF FP(N/J)+FP(D/J)=Ø
1.11 J=J-1
1.12 TO STEP 1.14 IF J<1
1.13 TO STEP 1.1Ø
1.14 PRINT A,"/",B,"+",C,"/",E,"=",N,"/",D
1.15 TO STEP 1.43 IF J<N
1.16 O=N/J
1.17 P=D/J
1.18 PRINT A,"/",B,"+",C,"/",E,"=",N,"/",D,"=",O,"/",P
1.19 TO STEP 1.43 IF O<P
1.2Ø TO STEP 1.41 IF FP(N/D)=Ø
1.21 L=1
1.22 K=N-L
1.23 L=L+1
1.24 TO STEP 1.26 IF FP(K/D)=Ø
1.25 TO STEP 1.22
1.26 Q=K/D
1.27 H=L-1
1.28 M=H
1.29 TO STEP 1.36 IF FP(H/M)+FP(D/M)=Ø
1.3Ø M=M-1
1.31 TO STEP 1.34 IF M<1
1.32 TO STEP 1.29
1.33 PRINT A,"/",B,"+",C,"/",E,"="
1.34 PRINT N,"/",D,"=",Q,"+",H,"/",D
1.35 TO STEP 1.43 IF M<H
1.36 X=H/M
1.37 Y=D/M
1.38 PRINT A,"/",B,"+",C,"/",E,"="
1.39 PRINT N,"/",D,"=",Q,"+",X,"/",Y
1.4Ø TO STEP 1.43 IF X<Y
1.41 F=N/D
1.42 PRINT A,"/",B,"+",C,"/",E,"=",N,"/",D,"=",F
1.43 PRINT "-"
```

Operation of Program:

A=2
B=3
C=5
E=6
2/3+5/6=9/6=1+1/2

A=1
B=4
C=3
D=2
1/4+3/2=7/4=1+3/4

A=4
B=7
C=2
D=5
4/7+2/5=34/35

A=1
B=9
C=1
D=6
7/9+5/6=29/18=1+11/18

PRM—A Program to Identify Prime Numbers

This program determines whether a number typed in by a user is a prime number, and if not it gives a factor. The program has been used by the sixth-graders at Winchester in connection with learning some elementary number theory. The program is amusing to the children because a number of comments are built in for various situations.

Listing of Program:

```
1.0 DEMAND N
1.001 SET P=0
1.01 SET END=1.67
1.02 DONE IF N=END
1.03 TO STEP 4.0 IF FP(N)>0
1.04 TO STEP 4.0 IF N<0
1.1 TO STEP 5.0 IF N<11
1.2 TO STEP 6.0 IF FP(N/2)=0
1.21 SET BBN=0
1.22 SET PDP=0
1.23 SAY HI
1.24 SAY  HO
1.3 DO PART 2 FOR I=3(2)SQRT(N)
1.35 TO STEP 1.0 IF P=1
1.4 SAY THIS NUMBER IS A PRIME
1.5 TO STEP 1.0

2.0 TO STEP 3.0 IF FP(N/I)=0

3.0 SAY THIS NUMBER IS NOT A PRIME
3.01 SAY  A FACTOR OF THIS NUMBER IS
3.02 SET FAC=I
3.03 TYPE FAC
3.04 LINE
3.1 LINE
3.2 SET P=1

4.0 SAY DON'T BE A WISE GUY
4.1 TO STEP 1.0

5.0 SAY NUMBERS OF ONE DIGIT DO NOT REQUIRE A COMPUTER
5.1 TO STEP 1.0

6.0 SAY THIS IS OBVIOUSLY AN EVEN NUMBER
6.1 TO STEP 1.0
```

Operation of Program:

```
N=3
NUMBERS OF ONE DIGIT DO NOT REQUIRE A COMPUTER
N=-89
DON'T BE A WISE GUY


N=56
THIS IS OBVIOUSLY AN EVEN NUMBER
N=65
HI
 HO
THIS NUMBER IS NOT A PRIME
 A FACTOR OF THIS NUMBER IS
          FAC =     5


N=91
HI
 HO
THIS NUMBER IS NOT A PRIME
 A FACTOR OF THIS NUMBER IS
          FAC =     7


N=13
HI
 HO
THIS NUMBER IS A PRIME
```

APPENDIX II

Assigned Problems and Demonstrations Used to Supplement Instruction

SECTION B: Ninth and Tenth Grade Programs

1. BASCHG - Change of Base
2. DECFRC - Changing a Repeating Decimal to a Fraction
3. LINEAR - Linear Curve Fitting
4. PRIMES - Generation of Primes
5. QUADEC - Solutions of Quadratics
6. SERIES - Terms in a Series

BASCHG - Change of Base

This program takes a base ten number and changes it to another base of the user's choice. Despite its inherent limitations, the program shows an understanding of what "bases" are. This was written by a 10th grader with no programming instruction at all. He merely watched others use the terminal and proceeded to write a working program on his own. Hence the program itself is quite cumbersome, but the thought behind it was obviously organized.

Listing of Program:

```
1.01 SAY THIS PROGRAM WILL CHANGE A NUMBER FROM BASE TEN
1.02 SAY TO A DESIGNATED BASE. BUT THE BASE MUST BE SMALLER
1.03 SAY THAN 10 AND THE NUMBER LESS THAN THE BASE TO THE
1.04 SAY SIXTH POWER.
1.05 LINE
1.06 SAY GIVE ME THE BASE (B) AND THE NUMBER(N).
1.1 DEMAND B
1.11 DEMAND N
1.12 LET A=N/B
1.13 LET C=IP(A)
1.14 LET D=C*B
1.15 LET E=N-D
1.16 LET F=C/B
1.17 LET G=IP(F)
1.18 LET H=G*B
1.19 LET I=C-H
1.2 LET J=G/B
1.21 LET K=IP(J)
1.22 LET L=K*B
1.23 LET M=G-L
1.24 LET O=K/B
1.25 LET P=IP(O)
1.26 LET Q=P*B
1.27 LET R=K-Q
1.28 LET S=P/B
1.29 LET T=IP(S)
```

```
1.3 LET U=T*B
1.31 LET V=P-U
1.32 LET W=T/B
1.33 LET X=IP(W)
1.34 LET Y=X*B
1.35 LET Z=T-Y
1.36 LET AB=X/B
1.37 LET AC=IP(AB)
1.38 LET AD=AC*B
1.39 LET AE=X-AD
1.4 LET AF=AE*10↑6
1.41 LET AG=Z*10↑5
1.42 LET AH=V*10↑4
1.43 LET AI=R*10↑3
1.44 LET AJ=M*10↑2
1.45 LET AK=I*10↑1
1.46 LET AL=E*10↑0
1.47 LET CN=AF+AG+AH+AI+AJ+AK+AL
1.48 TYPE CN
1.99 TO STEP 1.1
```

Operation of Program:

```
THIS PROGRAM WILL CHANGE A NUMBER FROM BASE TEN
TO A DESIGNATED BASE.  BUT THE BASE MUST BE SMALLER
THAN 10 AND THE NUMBER LESS THAN THE BASE TO THE
SIXTH POWER.

GIVE ME THE BASE (B) AND THE NUMBER(N).
B=3
N=45
            CN =  1200

B=5
N=134
            CN =  1014

B=7
N=567
            CN =  1440

B=9
N=23541
            CN =     3.5256*10↑4

B=2
N=38
            CN =     1.0011*10↑5
```

DECFRC - Changing a Repeating Decimal to a Fraction

This program was written by a student for use in teaching other students the technique of changing a repeating decimal to the fractional form of a rational number. As a side effect, it also forces him to reduce this fraction to lowest terms. This is interesting in that it represents a student's view on how a particular topic should be taught.

Listing of Program:

```
1.1 SAY GIVE ME THE ABC OF A REPEATING DECIMAL OF THE FORM
1.11 SAY .ABC... AND I WILL TEACH YOU TO REDUCE IT TO P/Q.
1.111 SAY (ANY NUMBER OF REPEATING DIGITS)
1.112 LINE
1.12 SAY TYPE IT TO 8 PLACES
1.2 DEMAND DEC
1.3 SAY HOW MANY DIGITS BEFORE IT REPEATS?
1.4 DEMAND NUM
1.5 SET P=IP(DEC*10↑NUM)
1.51 SET D=IP(DEC*10↑(2*NUM))
1.6 TO STEP 1.9 IF 'D-(P+P*(10↑NUM))'>.01
1.7 DO PART 2
1.8 DONE
1.9 SAY WRONG! NOW AGAIN
1.91 TO STEP 1.3

2.1 SAY TO SOLVE THIS PROBLEM:
2.2 SAY STEP ONE IS TO SUBTRACT DEC FROM DEC*10↑NUM
2.3 SAY STEP TWO IS TO RECOGNIZE THAT THIS IS THE P OF P/Q
2.4 SAY WHERE Q IS 10↑NUM -1
2.401 SAY THIS IS JUST SUBTRACTION
2.41 DO PART 4
2.5 SAY NOW LETS DO IT
2.51 TYPE DEC*10↑NUM
2.52 SET DEC=IP(DEC*10↑(7-NUM))
2.53 SET DEC=DEC/(10↑(7-NUM))
2.6 TYPE DEC
2.8 SET Q=(10↑NUM)-1
2.91 TYPE P,Q
2.92 DO PART 3
```

```
3.1 SET YES=1
3.2 SET NO=0
3.3 SAY THIS IS THE UNSIMPLIFIED ANSWER
3.4 DO PART 6 FOR A=2(1)P
3.5 SAY DEC CAN BE WRITTEN AS P/Q WHERE
3.6 TYPE DEC,P,Q
3.7 SAY CAN IT? LETS CHECK.
3.8 TYPE P/Q

4.01 LINE
4.1 SAY EXAMPLE: FOR DEC=.33...,NUM=1
4.11 LINE
4.2 SAY 10*DEC=3.333...
4.3 SAY   -DEC= .333...
4.31 SAY ---------------
4.4 SAY  9*DEC=3
4.5 SAY THEREFORE DEC=3/9
4.6 SAY WHICH CAN BE SIMPLIFIED TO 1/3.

6.1 SET Z=0
6.2 SAY ARE P AND Q BOTH DIVISIBLE BY A WHERE
6.3 TYPE A
6.4 DEMAND ANS
6.5 SET Z=1 IF 'FP(P/A)'<.01 AND 'FP(Q/A)'<.01
6.6 TO STEP 6.92 IF Z><ANS
6.7 DONE IF ANS=0
6.8 SET P=P/A
6.9 SET Q=Q/A
6.901 SAY P AND Q ARE NOW
6.902 TYPE P,Q
6.91 TO STEP 6.1
6.92 SAY YOU ARE WRONG
6.93 TO STEP 6.1
```

Operation of Program:

```
GIVE ME THE ABC OF A REPEATING DECIMAL OF THE FORM
.ABC... AND I WILL TEACH YOU TO REDUCE IT TO P/Q.
(ANY NUMBER OF REPEATING DIGITS)

TYPE IT TO 8 PLACES
DEC=.33333333
HOW MANY DIGITS BEFORE IT REPEATS?
NUM=1
TO SOLVE THIS PROBLEM:
STEP ONE IS TO SUBTRACT DEC FROM DEC*10↑NUM
STEP TWO IS TO RECOGNIZE THAT THIS IS THE P OF P/Q
WHERE Q IS 10↑NUM -1
THIS IS JUST SUBTRACTION
```

```
EXAMPLE: FOR DEC=.33...,NUM=1

10*DEC=3.333...
  -DEC=.333...
---------------
 9*DEC=3
THEREFORE DEC=3/9
WHICH CAN BE SIMPLIFIED TO 1/3.
NOW LETS DO IT
  DEC*10↑NUM =      3.333333

         DEC =       .333333

           P =      3
           Q =      9

THIS IS THE UNSIMPLIFIED ANSWER
ARE P AND Q BOTH DIVISIBLE BY A WHERE
           A =      2

ANS=NO
ARE P AND Q BOTH DIVISIBLE BY A WHERE
           A =      3

ANS=YES
P AND Q ARE NOW
           P =      1
           Q =      3

ARE P AND Q BOTH DIVISIBLE BY A WHERE
           A =      3

ANS=NO
DEC CAN BE WRITTEN AS P/Q WHERE
         DEC =       .333333
           P =      1
           Q =      3

CAN IT? LETS CHECK.

         P/Q =       .3333333
```

LINEAR - Linear Curve Fitting

This program computes a linear fit for two given sets of rectangular coordinates.

Listing of Program:

```
1.ØØ5 TYPE "YOU GIVE ME THE COORDINATES OF TWO POINTS,"
1.Ø1 TYPE "AND I WILL GIVE YOU AN EQUATION WHICH THOSE POINTS SATISFY"
1.Ø2 TYPE "P[1]=(X1,Y1),P[2]=(X2,Y2)"
1.Ø25 TYPE "NOW GIVE ME THE COORDINATES-ONE AT A TIME"
1.Ø26 TYPE #
1.Ø3 READ "P[1]=( ",B,", ",D," )  P[2]=( ",A,", ",C," )",#
1.Ø41 X=B,Y=D
1.Ø42 TYPE "YOU'VE GIVEN ME A POINT" IF B=A AND D=C
1.Ø43 TO STEP 1.45 IF B=A AND D=C
1.Ø45 PRINT"EQUATION IS X=  ",X IF B-A=Ø
1.Ø46 TO STEP 1.45 IF B-A=Ø
1.4 S=(D-C)/(B-A)
1.41 F=Y-(S*X)
1.42 PRINT"EQUATION IS Y="
1.43 PRINT S,"X+" IF S><Ø
1.435 SET F=Ø IF 'F'<1Ø↑-5
1.44 PRINT F
1.45 TYPE #,#,#
1.46 TO STEP 1.Ø3
```

Operation of Program:

```
←DO PART 1

YOU GIVE ME THE COORDINATES OF TWO POINTS,
AND I WILL GIVE YOU AN EQUATION WHICH THOSE POINTS SATISFY
P[1]=(X1,Y1),P[2]=(X2,Y2)
NOW GIVE ME THE COORDINATES-ONE AT A TIME

P[1]=( 3, 5 )  P[2]=( 1, 4 )
EQUATION IS  Y=.5X+3.5


P[1]=( -7, 12 )  P[2]=( Ø, Ø )
EQUATION IS  Y=-1.714286X+Ø


P[1]=( 1, 1 )  P[2]=( 1, 1 )
YOU'VE GIVEN ME A POINT


P[1]=( 12, 4 )  P[2]=( 12, 1 )
EQUATION IS X=  12


P[1]=( 1, 4 )  P[2]=( 2, 4 )
EQUATION IS  Y=4
```

PRIMES - Generation of Primes

This brief and elegant program will begin at two and type primes until it hits the limits of accuracy of the computer. Interesting problems were presented to the student in all of the following ways:

1. Show him a copy of the program and ask him what it does.
2. Show him the output of the program and ask him to write the program which does it.
3. Ask him to write a prime generating program containing only five steps.
4. Ask him to write a program which will operate more rapidly than this one when dealing with larger numbers (say greater than 200).

Listing of Program:

```
1.1 SET S=1, N=1
1.2 S=Ø IF FP(N/I)=Ø FOR I=2:1:SQRT(N)
1.3 TYPE N IF S=1
1.4 N=N+1, S=1
1.5 TO STEP 1.2
```

Operation of Program:

```
←DO PART 1
          N=     1
          N=     2
          N=     3
          N=     5
          N=     7
          N=    11
          N=    13
          N=    17
          N=    19
          N=    23
          N=    29
          N=    31
          N=    37
          N=    41
          N=    43
          N=    47
          N=    53
          N=    59
          N=    61
          N=    67
          N=    71
          N=    73
          N=    79
          N=    83
          N=    89
          N=    97
          N=   1Ø1
```

QUADEC - Solution of Quadratics

This program attempts to characterize particular solutions of the quadratic equation.

Listing of Program:

```
1.01 TYPE "THIS PROGRAM SOLVES EQUATIONS OF THE FORM"
1.02 TYPE "AX↑2+BX+C=0 FOR ANY VALUES OF A,B, AND C."
1.03 DEMAND A,B,C
1.04 TO STEP 1.17 IF A=0
1.05 D=B↑2-4*A*C
1.06 TO STEP 1.10 IF D<=0
1.07 TYPE"TWO REAL ROOTS: "
1.08 PRINT"R[1]=",(-B+SQRT(D))/(2*A),#
1.09 PRINT"R[2]=",(-B-SQRT(D))/(2*A),#,#
1.091 TO STEP 1.03
1.10 TO STEP 1.13 IF D<0
1.11 PRINT "ONE REAL ROOT: ",#,"R=",(-B/(2*A)),#,#
1.12 TO STEP 1.03
1.13 PRINT "TWO IMAGINARY ROOTS:",#
1.14 PRINT"R[1]=",-B/(2*A),"+",SQRT(-D)/(2*A),"I",#
1.15 PRINT"R[2]=",-B/(2*A),"-",SQRT(-D)/(2*A),"I",#,#
1.16 TO STEP 1.03
1.17 PRINT"ONE DEGREE EQUATION,ONE ROOT:",#,"R=",-C/B,#,# IF B<>0
1.19 TYPE"NO REAL OR IMAGINARY ROOTS",#,# IF B=0&C<>0
1.20 TO STEP 1.03 IF C<>0@B<>0
1.21 PRINT"IDENTITY, INDEPENDENT OF X",#,#
1.22 TO STEP 1.03
```

Operation of Program:

```
←DO PART 1
THIS PROGRAM SOLVES EQUATIONS OF THE FORM
AX↑2+BX+C=0 FOR ANY VALUES OF A,B, AND C.
          A=0
          B=5
          C=5
ONE DEGREE EQUATION,ONE ROOT:
R=-1

          A=1
          B=2
          C=1
ONE REAL ROOT:
R=-1

          A=4
          B=56
          C=12
TWO REAL ROOTS:
R[1]=-.21767
R[2]=-13.78233
```

SERIES—Guessing Terms in a Series

This student-written program helps another student discover the numbers used to generate a series in one of two ways. The program is both interesting and challenging to the user. It suggests that intuition isn't always enough to analyze even a simple-looking series. If a student feels he has a solution after seeing as many terms of the series as he desires, his solution is accepted and the series it would generate is produced. If he is incorrect he can request either more terms of the original series or some assistance in how to solve the problem.

Listing of Program:

```
1.01 SAY THIS PROGRAM PRODUCES A SERIES BY EITHER ADDING A
1.02 SAY CONSTANT TO A GIVEN NUMBER THEN MULTIPLYING BY ANOTHER
1.03 SAY CONSTANT, OR MULTIPLYING BY A CONSTANT THEN ADDING ANOTHER.
1.04 SAY THE OPERATOR IS TO DETERMINE THE TWO CONSTANTS USED
1.045 SAY AND THE ORDER OF OPERATIONS.
1.05 SET S=IP(RAN(2))
1.06 SET S=-1 IF S=0
1.07 SET AD=IP(RAN(11))*S
1.08 SET AD=0 IF IP(RAN(3))=1
1.09 SET MU=IP(RAN(11))*S
1.10 SET MU=0 IF IP(RAN(3))=1
1.11 SET F(1)=IP(RAN(3))
1.115 SET Y(1)=F(1)
1.12 SET YES=0
1.13 SET NO=1
1.14 SET PRT=IP(RAN(2))+2
1.15 TYPE F(1)
1.16 SET X=2
1.17 DO PART PRT
1.18 TYPE F(X)
1.185 SAY DO YOU WANT ANOTHER TERM?
1.19 DEMAND ANS
1.20 TO STEP 1.23 IF ANS=1
1.21 SET X=X+1
1.22 TO STEP 1.17
1.23 DO STEP 1.30
1.24 SAY IF NOT, MAYBE I CAN HELP YOU.
1.25 DEMAND ANS
```

```
1.26 TO STEP 1.35 IF ANS=0
1.27 SAY YOU NEED CONSIDER 3 ELEMENTS.
1.28 SAY IF X IS AN ELEMENT, THE NEXT ELEMENT IS
1.29 SAY IN THE FORM AX+B OR A(X+B).  SOLVE FOR A AND B.
1.30 SAY DO YOU HAVE A SOLUTION?
1.31 DEMAND ANS
1.32 TO STEP 1.35 IF ANS=0
1.33 SAY CONSULT YOUR INSTRUCTOR.
1.34 TO STEP 1.47
1.35 SAY IF ADD THEN MUL, ORD=2.  IF MUL THEN ADD, ORD=3
1.36 DEMAND ORD
1.37 DEMAND ADD
1.38 DEMAND MUL
1.385 SAY YOUR SERIES IS:
1.39 TYPE Y(1)
1.40 DO PART 4 FOR X=2,3 IF ORD=2
1.41 DO PART 5 FOR X=2,3 IF ORD=3
1.415 SET ORD=PRT IF ADD=0
1.42 TO STEP 1.49 IF ORD=PRT AND ADD=AD AND MUL=MU
1.425 TO STEP 1.49 IF Y(2)=F(2) AND Y(3)=F(3)
1.43 SAY TRY AGAIN?
1.44 DEMAND ANS
1.45 TO STEP 1.35 IF ANS=0
1.46 SAY ANSWER IS IN ORDER-------
1.47 TYPE AD,MU IF PRT=2
1.48 TYPE MU,AD IF PRT=3
1.49 SAY HAVE ANOTHER PROBLEM?
1.50 DEMAND ANS
1.51 TO STEP 1.05 IF ANS=0

2.1 SET F(X)=(F(X-1)+AD)*MU

3.1 SET F(X)=F(X-1)*MU+AD

4.1 SET Y(X)=(Y(X-1)+ADD)*MUL
4.2 DO PART 6

5.1 SET Y(X)=Y(X-1)*MUL+ADD
5.2 DO PART 6

6.1 SET Y(X)=0 IF 'Y(X)'<.000001
6.2 TYPE Y(X)
```

Operation of Program:

```
THIS PROGRAM PRODUCES A SERIES BY EITHER ADDING A
CONSTANT TO A GIVEN NUMBER THEN MULTIPLYING BY ANOTHER
CONSTANT, OR MULTIPLYING BY A CONSTANT THEN ADDING ANOTHER.
THE OPERATOR IS TO DETERMINE THE TWO CONSTANTS USED
AND THE ORDER OF OPERATIONS.
        F(1) =      1

        F(X) =     -2

DO YOU WANT ANOTHER TERM?
ANS=YES
        F(X) =     -2

DO YOU WANT ANOTHER TERM?
ANS=NO
DO YOU HAVE A SOLUTION?
IF NOT, MAYBE I CAN HELP YOU.
ANS=YES
IF ADD THEN MUL, ORD=2.  IF MUL THEN ADD, ORD=3
ORD=3
ADD=-2
MUL=0
YOUR SERIES IS:
        Y(1) =      1

        Y(X) =     -2

        Y(X) =     -2

HAVE ANOTHER PROBLEM?
ANS=YES
        F(1) =      1

        F(X) =     17

DO YOU WANT ANOTHER TERM?
ANS=YES
        F(X) =    161

DO YOU WANT ANOTHER TERM?
ANS=YES
        F(X) =   1457

DO YOU WANT ANOTHER TERM?
ANS=NO
DO YOU HAVE A SOLUTION?
IF NOT, MAYBE I CAN HELP YOU.
```

```
ANS=YES
IF ADD THEN MUL, ORD=2.  IF MUL THEN ADD, ORD=3
ORD=3
ADD=3
MUL=14
YOUR SERIES IS:
        Y(1) =      1

        Y(X) =     17

        Y(X) =    241

TRY AGAIN?
ANS=YES
IF ADD THEN MUL, ORD=2.  IF MUL THEN ADD, ORD=3
ORD=3
ADD=8
MUL=9
YOUR SERIES IS:
        Y(1) =      1

        Y(X) =     17

        Y(X) =    161

HAVE ANOTHER PROBLEM?
ANS=YES
        F(1) =      1

        F(X) =    -14

DO YOU WANT ANOTHER TERM?
ANS=YES
        F(X) =     61

DO YOU WANT ANOTHER TERM?
ANS=YES
        F(X) =   -314

DO YOU WANT ANOTHER TERM?
ANS=NO
DO YOU HAVE A SOLUTION?
IF NOT, MAYBE I CAN HELP YOU.
ANS=NO
YOU NEED CONSIDER 3 ELEMENTS.
IF X IS AN ELEMENT, THE NEXT ELEMENT IS
IN THE FORM AX+B OR A(X+B).  SOLVE FOR A AND B.
DO YOU HAVE A SOLUTION?
ANS=NO
CONSULT YOUR INSTRUCTOR.
          MU =     -5
          AD =     -9

HAVE ANOTHER PROBLEM?
```

APPENDIX II

Assigned Problems and Demonstrations Used to
Supplement Instruction

SECTION C: Eleventh and Twelfth Grade Programs

1. FUNCT - Guessing a Function
2. GRAPH - Plotting a Graph
3. SYNDIV - Synthetic Division
4. PI - Derivation of PI
5. DIOPHAN - Transfinite Numbers
6. FIBON - Fibonacci Numbers
7. ANALYZ - Solution Set of the General Quadratic
8. INEQ - Solution of Quadratic Inequalities
9. TRISOL - Solution of Triangles
10. SIMPAR - Simpson's Rule Integration
11. SIMTRP - Simpson's and Trapezoidal Rules
12. APPROX - Approximation to a Derivative

FUNCT—A Program which Guesses a Function Made Up by a Student

This program was written by Walter Koetke, a teacher in Lexington High School, for use in his 12th grade class. The program provided an effective incentive for students toward their own discovering and proving of the theorem that three non-collinear points uniquely determine a parabola.

Listing of Program:

```
1.01 SAY YOU MAKE UP A FUNCTION OF THE FORM Y=A*X↑2+B*X+C •
1.02 SAY I WILL THEN ASK YOU TO SUPPLY THE VALUE OF Y FOR
1.03 SAY CERTAIN VALUES OF X. AFTER VERY LITTLE OF THIS,
1.04 SAY I WILL GUESS THE FUNCTION YOU MADE UP.
1.05 SET YES=0
1.06 SET NO=1
1.065 SET R=1
1.07 LINE
1.08 SAY DO YOU HAVE A FUNCTION?
1.09 DEMAND ANS
1.095 LINE
1.10 TO STEP 1.13 IF ANS=0
1.11 SAY YOU'RE VERY UNIMAGINATIVE.
1.115 SAY LET SOMEONE ELSE TRY.
1.118 LINE
1.12 DONE
1.13 SET P=-1
1.14 SET Q=-1
1.15 SAY NOW SUPPLY THE VALUE OF Y WHEN
1.16 SET X=IP(RAN(10))+1
1.17 TO STEP 1.16 IF X=P OR X=Q OR X=0
1.18 SET P=X IF R=1
1.19 SET Q=X IF R=2
1.20 SET R=R+1
1.21 TYPE X
1.22 DEMAND Y
1.23 DO PART 50
1.24 LINE
1.25 TO STEP 1.15 IF R<4
1.26 DO PART 2 FOR K=1(1)3
1.27 SET A=D(1,4)
1.28 SET B=D(2,4)
1.29 SET C=D(3,4)
1.295 SAY YOUR EQUATION IS IN THE FORM
```

```
1.30 DO PART 6 IF A=0 AND B=0 AND C=0
1.31 DO PART 7 IF A=0 AND B=0 AND C><0
1.32 DO PART 8 IF A=0 AND B><0
1.33 DO PART 9 IF A><0
1.34 LINE
1.35 SAY AM I CORRECT?
1.36 DEMAND ANS
1.365 LINE
1.37 TO STEP 1.40 IF ANS=0
1.38 SAY I NEVER MISS (UNLESS YOU FUSS ABOUT SLIGHT ROUND-OFF ERROR)
1.385 SAY YOU'D BETTER CHECK YOUR COMPUTATION.
1.39 TO STEP 1.115
1.40 SAY GOOD, BUT I KNEW I WOULDN'T MISS.  NOW
1.41 TO STEP 1.115

2.01 DO PART 3 FOR J=4(-1 K
2.02 TO STEP 2.05 IF K=1
2.03 DO PART 4 FOR I=K-1(-1)1
2.04 DONE IF K=3
2.05 DO PART 4 FOR I=K+1(1)3

3.01 DET D(K,J)=D(K,J)/D(K,K)

4.01 DO PART 5 FOR J=4(-1)1

5.01 SET D(I,J)=D(I,J)-D(I,K)*D(K,J)
5.02 SET D(I,J)=0 IF 'D(I,J)'<.0001

6.01 SAY Y=0

7.01 SAY Y=C  WHERE
7.02 TYPE C

8.01 SAY Y=B*X+C  WHERE
8.02 TYPE B,C

9.01 SAY Y=A*X↑2+B*X+C  WHERE
9.02 TYPE A,B,C

50.1 SET D(R-1,1)=X↑2
50.2 SET D(R-1,2)=X
50.3 SET D(R-1,3)=1
50.4 SET D(R-1,4)=Y
```

Operation of Program:

```
YOU MAKE UP A FUNCTION OF THE FORM Y=A*X↑2+B*X+C .
I WILL THEN ASK YOU TO SUPPLY THE VALUE OF Y FOR
CERTAIN VALUES OF X.  AFTER VERY LITTLE OF THIS,
I WILL GUESS THE FUNCTION YOU MADE UP.

DO YOU HAVE A FUNCTION?
ANS=YES

NOW SUPPLY THE VALUE OF Y WHEN
          X =     10

Y=163

NOW SUPPLY THE VALUE OF Y WHEN
          X =      1

Y=19

NOW SUPPLY THE VALUE OF Y WHEN
          X =      3

Y=37

YOUR EQUATION IS IN THE FORM
Y=A*X↑2+B*X+C   WHERE
          A =      1
          B =      5
          C =     13


AM I CORRECT?
ANS=YES

GOOD, BUT I KNEW I WOULDN'T MISS.  NOW
LET SOMEONE ELSE TRY.

←
```

GRAPH - Plots the Graph of a Specified Function Y = F(X) for Specified Lower Bound and Increment in the Argument X and Specified Scale in Y.

James Hudson, a student at Lexington High School, is the author. The program is one of several written at various schools for calculating points for subsequent plotting and for actually developing a graph, as in the present case. The interest in a graphing capability is, clearly, general. This particular program incorporates clever scaling and rounding features, and does relatively fast plotting.

Listing of Program:

```
1.01 SET X=0
1.02 SET X=0
1.03 SAY BY TYPING "2.05 SET Y=..." THEN "DO PART 2"
1.1 SAY THIS PLOTS VALUE OF FUNCTIONS OF X. YOU TYPE IN YOUR
1.2 SAY EQUATION "2.05 SET Y=..." AND THEN "DO PART 2." XMI IS THE
1.3 SAY MIM(X), SLX TH E X STEP LENGTH, AND SLY, THE Y

2.001 DEMAND XMI
2.002 DEMAND SLX
2.003 DEMAND SLY
2.0035 SET XMA=XMI+30*SLX
2.004 PAGE
2.01 SET X=XMI
2.02 DO PART 35 IF XMI>0 OR XMA<0
2.03 SAY                X
2.04 DO PART 35 IF X=0
2.05 SET Y=X↑2-15
2.06 TO STEP 2.10 IF X=0 OR Y>15*SLY OR Y<-15*SLY
2.07 SET Q=SGN(Y)
2.075 SET K=Q*IP('Y/SLY'+.001)
2.08 DO PART K+18 IF 'Y/SLY-K'<=SLY/2
2.09 DO PART K+19 IF 'Y/SLY-K'>SLY/2
2.10 SET Q=SGN(X)
2.11 DO PART 18 IF Y>15*SLY OR Y<-15*SLY
2.12 SET X=X+SLX IF X>=0
2.13 SET X=Q*('X'-SLX) IF X<0
```

```
2.14 SET X=0 IF 'X'<.0001
2.15 DONE IF X>XMA
2.16 TO STEP 2.04

3.0 SAY  .              .

4.0 SAY   .             .

5.0 SAY    .            .

6.0 SAY     .           .

7.0 SAY      .          .

8.0 SAY       .         .

9.0 SAY        .        .

10. SAY         .       .

11. SAY          .      .

12. SAY           .     .

13. SAY            .    .

14. SAY             .   .

15. SAY              .  .

16. SAY               . .

17. SAY                ..

18. SAY                 .

19. SAY                 ..

20. SAY                 . .

21. SAY                 .  .

22. SAY                 .   .

23. SAY                 .    .

24. SAY                 .     .

25. SAY                 .      .
```

26. SAY • •

27. SAY • •

28. SAY • •

29. SAY • •

30. SAY • •

31. SAY • •

32. SAY • •

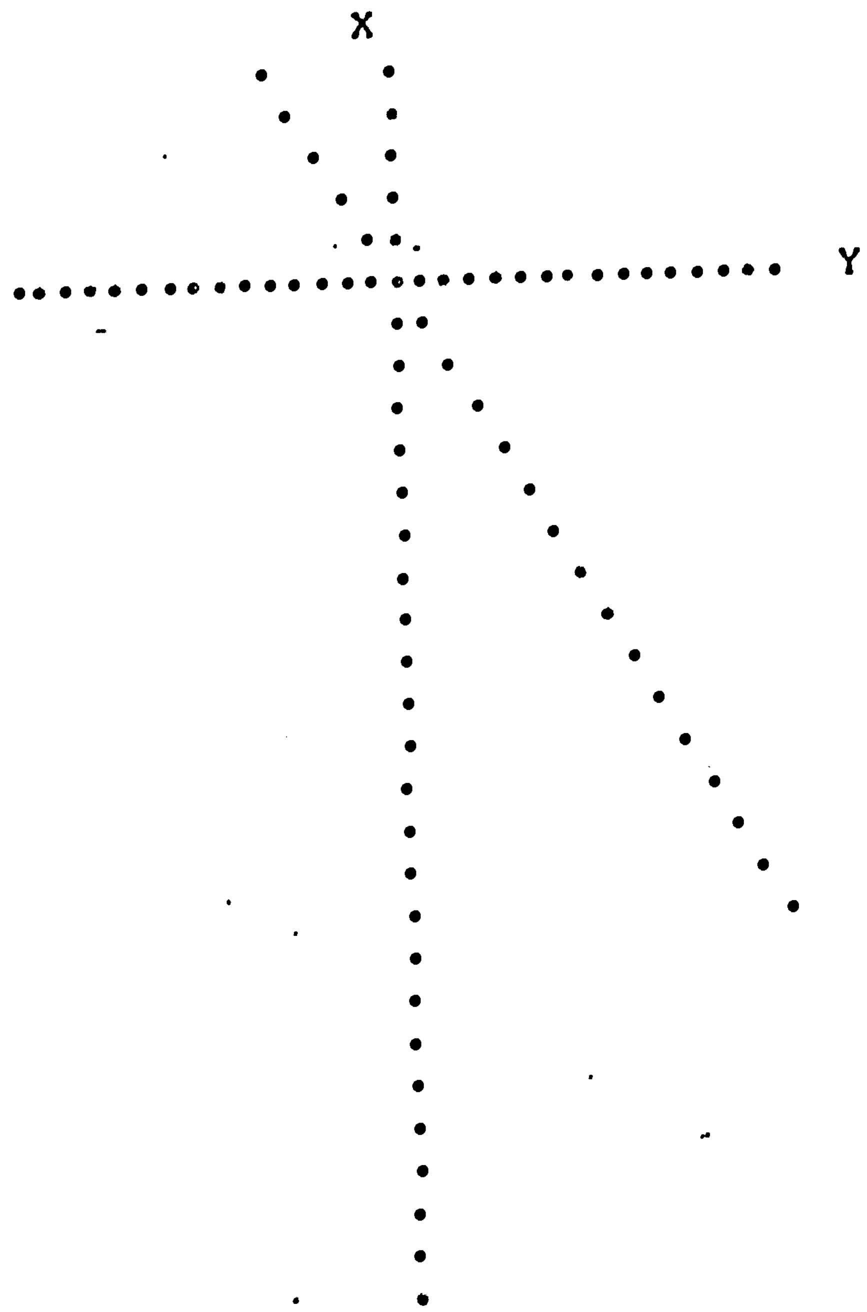
33. SAY • •

35. SAY .... Y

Operation of Program:

```
DO PART 1

BY TYPING "2.05 SET Y=..." THEN "DO PART 2"
THIS PLOTS VALUE OF FUNCTIONS OF X. YOU TYPE IN YOUR
EQUATICN "2.05 SET Y=..." AND THEN "DO PART 2." XMI IS THE
MIM(X), SLX TH E X STEP LENGTH, AND SLY, THE Y
←2.05 SET Y=X
←DO PART 2
XMI=-5
SLX=1
SLY=1
```



SYNDIV - Synthetic Division

This program will divide a polynomial of the form $a(o)x^n + a(1)x^{n-1} + \ldots\ldots + a(n)$ by x-(div) synthetically, a method which can be used to factor the polynomial, or to find or approximate its zeros. It is a very convenient program to have on hand if a large amount of synthetic division needs to be done, as this is tedious to perform by hand. In addition, the program involves very little programming technique, and closely approximates the doing of the division itself in logical structure. Asking the student to write it should serve to fix a firm understanding of the method of synthetic division.

Listing of Program:

```
1.Ø LINE
1.1 TYPE "GIVE ME THE COEFFICIENTS OF THE N DEGREE POLYNOMIAL P IN"
1.2 TYPE "SEQUENCE FROM A[N] TO A[Ø] AND I WILL DETERMINE THE"
1.3 TYPE "COEFFICIENTS OF THE QUOTIENT RESULTING FROM DIVISION OF"
1.4 TYPE "P BY THE LINEAR FACTOR, (X-F)."
1.5 TO PART 2

2.1 LINE
2.2 DEMAND N
2.3 DO PART 3 FOR T=N:-1:Ø
2.4 DEMAND F
2.45 LINE
2.5 Q[N]=A[N]
2.6 DO PART 4 FOR T=N:-1:1
2.7 N←N-1
2.8 DO PART 5 FOR T=Ø:1:N
2.9 DO PART 6

3.1 DEMAND A[T]

4.1 Q[T-1]=Q[T]*F+A[T-1]

5.1 A[T]=Q[T+1]

6.1 DO PART 7 FOR T=N:-1:Ø
6.2 PRINT "THE REMAINDER IS "
6.3 TO PART 8 IF 'Q[Ø]'+1/2+''Q[Ø]'-1/2'=1
6.4 PRINT Q[Ø]

7.1 TYPE A[T]

8.1 PRINT "Ø"

←
```

Operation of Program:

```
DO PART 1

GIVE ME THE COEFFICIENTS OF THE N DEGREE POLYNOMIAL P IN
SEQUENCE FROM A[N] TO A[Ø] AND I WILL DETERMINE THE
COEFFICIENTS OF THE QUOTIENT RESULTING FROM DIVISION OF
P BY THE LINEAR FACTOR, (X-F).

          N=4
       A[4]=1
       A[3]=1
       A[2]=1
       A[1]=1
       A[Ø]=1
          F=1

       A[3]=     1
       A[2]=     2
       A[1]=     3
       A[Ø]=     4
THE REMAINDER IS 5
←
```


3.

PI - Derivation of PI

This program follows the classical derivation of PI using the method of Archimedes. It finds the perimeters of the inscribed and circumscribed polygons of a specified number of sides (N), for a circle of specified radius. Using these parameters, it relates the values of the perimeters to the radius of the circle, and approximates the value of PI for each value of N. The program demonstrates in a very effective manner the concepts of convergence and limits.

Listing of Program:

```
1.001 TYPE #,"GIVE ME THE RADIUS (R) OF A CIRCLE AND I WILL"
1.002 TYPE "APPROXIMATE ITS CIRCUMFERENCE, AND THE VALUE OF"
1.003 TYPE "PI, BY FINDING THE PERIMETER (PA) OF AN INSCRIBED"
1.004 TYPE "REGULAR POLYGON OF N SIDES.",#
1.006 TYPE "I WILL ALSO FIND THE PERIMETER (PB) OF A "
1.007 TYPE "CIRCUMSCRIBED REGULAR POLYGON OF THE SAME"
1.008 TYPE "NUMBER OF SIDES.",#
1.009 TYPE "YOU MUST TELL ME INITIALLY THE NUMBER OF SIDES (N)"
1.010 TYPE "AND I WILL THEN PROCEED BY SUCCESSIVELY DOUBLING"
1.011 TYPE "THAT NUMBER.",#
1.012 TYPE "YOU MUST SUPPLY BOTH N AND R WHEN REQUESTED.",#,#,#
1.016 TO PART 2

2.0 DEMAND R,N
2.02 E=R*SQRT(3),F=2*E IF N=3
2.03 E=R*SQRT(2), F=2*R IF N=4
2.04 DEMAND E,F IF N<>3 OR N<>4
2.05 TO PART 3

3.10 PA=N*E,PB=N*F
3.12 TYPE N,PA,PB,PA/(2*R),PB/(2*R),(PA+PB)/(4*R)
3.13 SET N=2*N
3.14 DONE IF N>5000
3.15 SET E=SQRT(2*R↑2-R*SQRT(4*R↑2-E↑2))
3.16 SET F=(2*R*SQRT(F↑2+4*R↑2)-4*R↑2)/F
3.17 TO STEP 3.10
```

Operation of Program:

```
 DO PART 1

GIVE ME THE RADIUS (R) OF A CIRCLE AND I WILL
APPROXIMATE ITS CIRCUMFERENCE, AND THE VALUE OF
PI, BY FINDING THE PERIMETER (PA) OF AN INSCRIBED
REGULAR POLYGON OF N SIDES.

I WILL ALSO FIND THE PERIMETER (PB) OF A
CIRCUMSCRIBED REGULAR POLYGON OF THE SAME
NUMBER OF SIDES.

YOU MUST TELL ME INITIALLY THE NUMBER OF SIDES (N)
AND I WILL THEN PROCEED BY SUCCESSIVELY DOUBLING
THAT NUMBER.

YOU MUST SUPPLY BOTH N AND R WHEN REQUESTED.


          R=1
          N=4

          N=     4
         PA=     5.656
         PB=     8
    PA/(2*R)=    2.828
    PB/(2*R)=    4
(PA+PB)/(4*R)=    3.414
          N=     8
         PA=     6.121819
         PB=     6.627417
    PA/(2*R)=    3.060909
    PB/(2*R)=    3.313709
(PA+PB)/(4*R)=    3.187309
          N=    16
         PA=     6.241705
         PB=     6.365196
    PA/(2*R)=    3.120853
    PB/(2*R)=    3.182598
(PA+PB)/(4*R)=    3.151725
          N=    32
         PA=     6.271894
         PB=     6.303453
    PA/(2*R)=    3.135947
    PB/(2*R)=    3.151726
(PA+PB)/(4*R)=    3.143837
```

Operation of Program: (continued)

```
              N=   64
             PA=     6.279452
             PB=     6.288251
      PA/(2*R)=      3.139726
      PB/(2*R)=      3.144126
(PA+PB)/(4*R)=        3.141926
              N=  128
             PA=     6.281347
             PB=     6.284429
      PA/(2*R)=      3.140674
      PB/(2*R)=      3.142214
(PA+PB)/(4*R)=        3.141444
              N=  256
             PA=     6.281794
             PB=     6.283512
      PA/(2*R)=      3.140897
      PB/(2*R)=      3.141756
(PA+PB)/(4*R)=        3.141326
              N=  512
             PA=     6.281794
             PB=     6.283807
      PA/(2*R)=      3.140897
      PB/(2*R)=      3.141904
(PA+PB)/(4*R)=        3.1414
```

DIOPHAN - Transfinite Numbers

This program was the result of a discussion in an advanced placement class on transfinite numbers. The teacher was attempting to show that $\aleph_0 \cdot \aleph_0 = \aleph_0$ and had arranged the set NXN in sequence as follows:

(1,1); (1,2), (2,1); (1,3), (2,2), (3,1); (1,4), (2,3), (3,2), (4,1); ... .

While it was fairly apparent that given (M,N) one could determine its index in the sequence, the class was unconvinced that given L, the index, one could always tell what the ordered pair would be. Hence, the computer was used to solve the diophantine equation:

(M+N-2)(M+N-1)/2 +M =L

Listing of Program:

```
1.1 TYPE "GIVE ME A VALUE FOR L, AND I WILL FIND THE POSITIVE INTEGRAL"
1.2 TYPE "SOLUTIONS FOR M AND N WITH SMALLEST M, SATISFYING THE"
1.3 TYPE "EQUATION:  (M+N-2)(M+N-1)/2 +M =L.",#,#
1.4 DEMAND L
1.5 M=Ø
1.6 M=M+1
1.7 N=(-(2*M-3)+SQRT(-8*M+1+8*L))/2
1.8 TO PART 4 IF IP(N)=N
1.9 TO STEP 1.6

4.1 TYPE M,N
4.2 DONE
```

Operation of Program:

```
←DO PART 1
GIVE ME A VALUE FOR L, AND I WILL FIND THE POSITIVE INTEGRAL
SOLUTIONS FOR M AND N WITH SMALLEST M, SATISFYING THE
EQUATION:  (M+N-2)(M+N-1)/2 +M =L.

           L=6
           M=     3
           N=     1
←DO PART 1.4
           L=17
           M=     2
           N=     5
```

FIBON - Fibonacci Numbers

This program generates a Fibonacci series with standard starting values, 1 and 1, or with starting values specified by the student.

Listing of Program:

```
1.001 XX=2
1.002 YES=1,NO=2,R=1,M=2,P=1,A=1
1.003 READ "DO YOU WANT THE REGULAR FIBONACCI SERIES? ",F,#
1.004 TO STEP 1.2 IF F=2
1.005 READ "WHAT SHALL I USE AS UPPER LIMIT FOR THIS SERIES? ",D,#
1.02 TYPE N[P] FOR N[P]=A
1.03 A=N[P] +N[P-1],P=P+1
1.035 TO STEP 1.2 IF A>D
1.04 TO STEP 1.02
1.05 R=1,M=2,P=1,A=1
1.2 X=IP(RAN(10))+1,Y=IP(RAN(10))+1
1.3 PRINT "NOW USE FIRST TERMS ",X," AND ",Y," TO GENERATE A SERIES",#
1.31 Q[M-1]=MAX(X,Y),Q[M-2]=MIN(X,Y)
1.32 READ "DO YOU WANT TO SEE ALL TERMS OF THIS SERIES? ",G,#
1.33 TYPE Q[M-2],Q[M-1] IF G=1
1.4 Q[M]=Q[M-2]+Q[M-1]
1.41 TYPE Q[M] IF G=1
1.42 M=M+1
1.91 C[R]=Q[M-2]/Q[M-1]
1.919 TO STEP 1.4 IF FP((M-2)/5)><0
1.92 TYPE # IF G=1
1.921 PRINT "RATIO OF TERM ",M-2," OVER TERM ",M-1," IS ",C[R],#,#
1.95 READ "WOULD YOU LIKE ANOTHER SERIES? ",XX IF 'C[R]-C[R-1]'<.00001
1.9501 TYPE #
1.951 DONE IF XX=NO AND 'C[R]-C[R-1]'<.00001
1.952 TO STEP 1.05 IF XX=YES AND 'C[R]-C[R-1]'<.00001
1.954 R=R+1
1.96 TO STEP 1.4
```

Operation of Program:

```
DO YOU WANT THE REGULAR FIBONACCI SERIES? YES
WHAT SHALL I USE AS UPPER LIMIT FOR THIS SERIES? 50
       N[1]=     1
       N[2]=     1
       N[3]=     2
       N[4]=     3
       N[5]=     5
       N[6]=     8
       N[7]=    13
       N[8]=    21
       N[9]=    34
NOW USE FIRST TERMS 10 AND 5 TO GENERATE A SERIES
DO YOU WANT TO SEE ALL TERMS OF THIS SERIES? YES
       Q[0]=     5
       Q[1]=    10
       Q[2]=    15
       Q[3]=    25
       Q[4]=    40
       Q[5]=    65
       Q[6]=   105

RATIO OF TERM 5 OVER TERM 6 IS .6190476


       Q[7]=   170
       Q[8]=   275
       Q[9]=   445
      Q[10]=   720
      Q[11]=  1165

RATIO OF TERM 10 OVER TERM 11 IS .6180257


      Q[12]=  1885
      Q[13]=  3050
      Q[14]=  4935
      Q[15]=  7985
      Q[16]=     1.292*10↑4

RATIO OF TERM 15 OVER TERM 16 IS .6180341

WOULD YOU LIKE ANOTHER SERIES? NO
←
```

ANALYZ—A Solution Set of the General Quadratic

This program takes a quadratic equation, examines the constants, and decides whether the equation represents a circle, line, parabola, etc.

In writing this kind of program, the student gains practice in analyzing particular cases in the context of the general case, meanwhile acquiring a thorough understanding of the subject.

Listing of Program:

```
1.11 SAY THIS PROGRAM WILL DESCRIBE THE SOLUTION SET OF THE
1.12 SAY GENERAL QUADRATIC EQUATION AX↑2+CY↑2+DX+EY+F=0.
1.13 SET YES=1
1.14 SET NO=0
1.15 LINE
1.16 SAY DO YOU HAVE AN EQUATION?
1.17 DEMAND ANS
1.19 LINE
1.20 DONE IF ANS=0
1.21 SAY GOOD.  NOW SUPPLY THE CONSTANTS.
1.22 DEMAND A
1.23 DEMAND C
1.24 DEMAND D
1.25 DEMAND E
1.26 DEMAND F
1.27 LINE
1.28 TO STEP 2.11
1.29 LINE
1.30 LINE
1.31 SAY DO YOU HAVE ANOTHER EQUATION?
1.32 TO STEP 1.17

2.11 TO STEP 4.11 IF 'A'+'C'+'D'+'E'+'F'=0
2.12 TO STEP 4.13 IF 'A'+'C'+'D'+'E'=0
2.13 TO STEP 4.15 IF 'A'+'C'=0
2.14 TO STEP 2.21 IF A><C
2.15 SET Q=-F/A+(D↑2+E↑2)/(4*A↑2)
2.16 TO STEP 2.18 IF 'Q'>.00001
2.17 SET Q=0
2.18 TO STEP 4.17 IF Q>0
2.19 TO STEP 4.19 IF Q=0
2.20 TO STEP 4.21
```

```
2.21 TO STEP 2.29 IF A><0
2.22 TO STEP 4.23 IF D><0
2.23 SET Q=E↑2-4*C*F
2.24 TO STEP 2.26 IF 'Q'>.00001
2.25 SET Q=0
2.26 TO STEP 4.25 IF Q>0
2.27 TO STEP 4.15 IF Q=0
2.28 TO STEP 4.21
2.29 TO STEP 2.33 IF C><0
2.30 TO STEP 4.23 IF E><0
2.31 SET Q=D↑2-4*A*F
2.32 TO STEP 2.24
2.33 SET Q=(D↑2/(4*A)+E↑2/(4*C)-F)/(A*C)
2.34 TO STEP 2.36 IF 'Q'>.00001
2.35 SET Q=0
2.36 TO STEP 2.39 IF SGN(A)=SGN(C)
2.37 TO STEP 4.25 IF Q=0
2.38 TO STEP 4.29
2.39 TO STEP 4.21 IF SGN(A)><SGN(Q)
2.40 TO STEP 4.21 IF Q=0
2.41 TO STEP 4.27

3.32 LINE

4.11 SAY EQUATION REPRESENTS A PLANE
4.12 TO STEP 1.29
4.13 SAY NO SOLUTIONS TO EQUATION
4.14 TO STEP 1.29
4.15 SAY EQUATION REPRESENTS A LINE
4.16 TO STEP 1.29
4.17 SAY EQUATION REORESENTS A CIRCLE
4.18 TO STEP 1.29
4.19 SAY EQUATION REPRESENTS A POINT CIRCLE
4.20 TO STEP 1.29
4.21 SAY NO REAL SOLUTIONS TO EQUATION
4.22 TO STEP 1.29
4.23 SAY EQUATION REPRESENTS A PARABOLA
4.24 TO STEP 1.29
4.25 SAY EQUATION REPRESENTS TWO LINES
4.26 TO STEP 1.29
4.27 SAY EQUATION REPRESENTS AN ELLIPSE
4.28 TO STEP 1.29
4.29 SAY EQUATION REPRESENTS A HYPERBOLA
4.30 TO STEP 1.29
```

Operation of Program:

```
THIS PROGRAM WILL DESCRIBE THE SOLUTION SET OF THE
GENERAL QUADRATIC EQUATION AX↑2+CY↑2+DX+EY+F=Ø.

DO YOU HAVE AN EQUATION?
ANS=YES

GOOD.  NOW SUPPLY THE CONSTANTS.
A=5
C=-5
D=8
E=-13
F=2

EQUATION REPRESENTS A HYPERBOLA


DO YOU HAVE ANOTHER EQUATION?
ANS=YES

GOOD.  NOW SUPPLY THE CONSTANTS.
A=5
C=8
D=3
E=6
F=9

NO REAL SOLUTIONS TO EQUATION


DO YOU HAVE ANOTHER EQUATION?
ANS=YES

GOOD.  NOW SUPPLY THE CONSTANTS.
A=5
C=-5
D=Ø
E=Ø
F=Ø

EQUATION REPRESENTS TWO LINES

DO YOU HAVE ANOTHER EQUATION?
ANS=NO
```

INEQ - A Program to Solve Quadratic Inequalities of the Form $AX^2 + BX + C < K$ or $> K$.

This program was written by Walter Koetke, a teacher in Lexington High School, for use in his 12th grade class. It was motivated by the success with the program FUNCT. That program was effective in encouraging students to prove a theorem by themselves reproducing a surprising, and initially baffling, TELCOMP program. About half of Mr. Koetke's students were enabled to program solutions for general quadratic inequalities after seeing the following program in demonstration, without any help. Another quarter of the class were able to solve the problem with minimal help. The reader should note that problems in inequalities are among the most difficult for high school students.

Listing of Program:

```
1.01 SAY I WILL SOLVE QUADRATIC INEQUALITIES OF THE FORM
1.02 SAY AX↑2+BX+C><K.
1.03 SAY I WILL DEMAND A,B,C,K, AND INQ. TYPE IN THE VALUES
1.04 SAY OF THE CONSTANTS. TYPE G IF YOUR INEQUALITY IS
1.05 SAY GREATER THAN AND L IF INEQUALITY IS LESS THAN.
1.06 LINE
1.07 SET G=1
1.08 SET L=-1
1.09 DEMAND A
1.10 DEMAND B
1.11 DEMAND C
1.12 DEMAND K
1.13 DEMAND INQ
1.135 SAY SOLUTIONS TO YOUR INEQUALITY ARE
1.14 SET A=A*INQ
1.16 SET B=B*INQ
1.17 SET C=(C-K)*INQ
1.18 TO STEP 1.50 IF A><0
1.19 TO STEP 1.40 IF B=0
1.20 SET P=-C/B
```

```
1.21 DO PART 15 IF B>0
1.22 DO PART 16 IF B<0
1.23 TO STEP 1.06
1.40 DO PART 11 IF C>0
1.41 DO PART 17 IF C<0
1.42 TO STEP 1.06
1.50 SET D=B↑2-4*A*C
1.505 SET D=0 IF 'D'<.0001
1.51 TO STEP 1.54 IF D<0
1.52 SET P=(-B+SQRT(D))/(2*A)
1.53 SET Q=(-B-SQRT(D))/(2*A)
1.54 TO STEP 1.55
1.55 DO PART 10 IF A<0 AND D<=0
1.56 DO PART 11 IF A>0 AND D<0
1.57 DO PART 12 IF A>0 AND D=0
1.58 DO PART 14 IF D>0
1.59 TO STEP 1.06
1.60 DONE

10.0 SAY NOT REAL.

11.0 SAY ALL REAL VALUES OF X.

12.0 SAY ALL REAL VALUES OF X EXCEPT P, WHERE
12.1 TYPE P

14.0 SAY ALL REAL VALUES OF X GREATER THAN P AND LESS THAN Q, WHERE
14.1 TYPE P,Q

15.0 SAY ALL REAL VALUES OF X GREATER THAN P, WHERE
15.1 TYPE P

16.0 SAY ALL REAL VALUES OF X LESS THAN P, WHERE
16.1 TYPE P

17.0 NONEXISTANT.

←
```

Operation of Program:

```
I WILL SOLVE QUADRATIC INEQUALITIES OF THE FORM
AX↑2+BX+C><K.
I WILL DEMAND A,B,C,K, AND INQ.  TYPE IN THE VALUES
OF THE CONSTANTS.  TYPE G IF YOUR INEQUALITY IS
GREATER THAN AND L IF INEQUALITY IS LESS THAN.
```

```
A=2
B=5
C=12
K=9
INQ=G
SOLUTIONS TO YOUR INEQUALITY ARE
ALL REAL VALUES OF X GREATER THAN P AND LESS THAN Q, WHERE
           P =     -1
           Q =     -1.5


A=-3
B=6
C=13
K=-5
INQ=L
SOLUTIONS TO YOUR INEQUALITY ARE
ALL REAL VALUES OF X GREATER THAN P AND LESS THAN Q, WHERE
           P =     3.645751
           Q =    -1.645751


A=-3
B=-6
C=-13
K=5
INQ=G
SOLUTIONS TO YOUR INEQUALITY ARE
NOT REAL.

A=-3
B=-6
C=-13
K=5
INQ=L
SOLUTIONS TO YOUR INEQUALITY ARE
ALL REAL VALUES OF X.
```

TRISOL - A Program for "Solving" Triangles.

This ambitious project of "JPR and CAE" at Phillips Academy, Andover, calculates the unknown sides or angles, and the area, given information on known sides or angles. It considers five cases: all three sides known (SSS), two sides with enclosed angle known (SAS), etc. In the (ASS) case it evaluates both solutions.

Listing of Program:

```
1.0 SAY M=1 IF SSS; M=2 IF SAS; M=3 IF SAA;
1.001 SAY M=4 IF ASA; M=5 IF ASS.
1.1 DEMAND M
1.2 DO PART 2 IF M=1
1.3 DO PART 3 IF M=2
1.4 DO PART 4 IF M=3 OR M=4
1.45 DO PART 7 IF M=5
1.5 SET AR=A*B*SIN(LC*.01745329)/2
1.6 TYPE AR
1.7 TO STEP 1.1

2.0 DEMAND A
2.1 DEMAND B
2.2 DEMAND C
2.25 SET S=(A+B+C)/2
2.3 SET R=SQRT((S-A)*(S-B)*(S-C)/S)
2.4 SET LA=ARG(R/(S-A))*114.59158
2.5 SET LB=ARG(R/(S-B))*114.59158
2.6 SET LC=ARG(R/(S-C))*114.59158
2.7 TYPE LA,LB,LC

3.0 DEMAND LA
3.1 DEMAND B
3.2 DEMAND C
3.3 SET LA=LA*.01745329
3.4 SET A=SQRT(B↑2+C↑2-2*B*C*(COS(LA)))
3.5 SET X=(3.141593-LA)/2
3.6 SET Y=ARG((B-C)*SIN(X)/((B+C)*COS(X)))
3.7 SET LB=(X+Y)*57.29579
3.8 SET LC=(X-Y)*57.29579
3.9 TYPE A,LB,LC
```

```
4.0 DEMAND LA
4.1 DEMAND LB
4.2 SET LA=LA*.01745329
4.3 SET LB=LB*.01745329
4.4 SET LC=[3.141593-LA-LB]
4.5 DO PART 5 IF M=3
4.6 DO PART 6 IF M=4

5.0 DEMAND A
5.1 SET B=(A*SIN(LB))/SIN(LA)
5.2 SET C=(A*SIN(LC))/SIN(LA)
5.3 SET LC=LC*57.29579
5.4 TYPE B,C,LC

6.0 DEMAND C
6.1 SET A=C*SIN(LA)/SIN(LC)
6.2 SET B=C*SIN(LB)/SIN(LC)
6.3 SET LC=LC*57.29579
6.4 TYPE A,B,LC

7.00 DEMAND A
7.01 DEMAND B
7.02 DEMAND LA
7.03 SET LA=LA*.01745329
7.04 SET X=B*SIN(LA)
7.05 TO STEP 9.0 IF X>A
7.06 SET LB=B*SIN(LA)/A
7.07 SAY LB=SIN(LB); GIVE THE RADIAN OF LB
7.08 TYPE LB
7.09 DEMAND RAD
7.10 SET LB=RAD
7.11 DO PART 8
7.12 DO PART 8 FOR LB=[3.141593-RAD]

8.0 SET LC=[3.141593-LA-LB]
8.1 SET C=SIN(LC)*A/SIN(LA)
8.2 SET LB=LB*57.29579
8.3 SET LC=LC*57.29579
8.4 TYPE LB,LC,C

9.0 SAY NO SOLUTION
9.1 DONE
```

Operation of Program:

```
M=1 IF SSS; M=2 IF SAS; M=3 IF SAA;
M=4 IF ASA; M=5 IF ASS.
M=1
A=1
B=2
C=SQRT(3)
                LA =     30
                LB =     90.00002
                LC =     60.00001

                AR =       .8660254

M=2
LA=45
B=5
C=2
                 A =      3.85459
                LB =    113.4761
                LC =     21.52399

                AR =      3.535533

M=3
LA=60
LB=45
A=3
                 B =      2.44949
                 C =      3.346066
                LC =     75.00005

                AR =      3.549039

M=4
LA=45
LB=35
C=2
                 A =      1.43603
                 B =      1.16485
                LC =    100

                AR =       .8236729
```

```
M=5
A=4
B=4
LA=45
LB=SIN(LB); GIVE THE RADIAN OF LB
            LB =       .7071067

RAD=1
            LB =     57.29579
            LC =     77.70426
             C =      5.527094

            LB =    122.7043
            LC =     12.29579
             C =      1.20475

            AR =      1.703668
```

SIMPAR - A Program for Simpson's Rule Integration of a Specified Function

Simpson's rule is one of the simpler methods for evaluating the definite integral of a function y=f(x) from s=a to s=b. It is derived by dividing b-a into subintervals, and then constructing parabolas through each group of three values of the function at consecutive intervals. The sum of the areas under these paraboli will approximate the definite integral, and will be dx/3(f(a) + 4f(a+dx) + 2f(a+2dx) + 4f(a+3dx) + 2f(a+4dx) +......+ 4f(b-dx) + f(b)). The program merely evaluates this expression for a given f(x) and given values of a, b, and N (the number of intervals). It incorporates, in a straightforward way, options permitting the user to modify for a given integrand the interval of integration or the end points.

Listing of Program:

```
31.Ø5 TYPE #,#,#
31.1 TYPE " I WILL DETERMINE BY SIMPSON'S RULE THE AREA"
31.2 TYPE " UNDER THE CURVE Y=F(X) BETWEEN A AND B.  PLEASE NOTE"
31.3 TYPE " THAT I WILL ASK YOU FOR THE VALUES OF A AND B AND ALSO FOR"
31.4 TYPE "N, THE NUMBER OF SLICES YOU WANT ME TO MAKE."
31.5 TYPE " N MUST BE AN EVEN INTEGER."
31.6 TO PART 32

32.Ø TYPE "YOU MUST TYPE THE COMMAND:  36.Ø SET Y=....."
32.1 TYPE " AND THEN TYPE: DO PART 33"

33.1 DEMAND A
33.2 DEMAND B
33.3 TO PART 34

34.1 DEMAND N
34.2 TO PART 37 IF 'FP(N/2)' >1/4
34.3 TO PART 35

35.Ø1 SET H='B-A'/N
35.Ø2 SET T=Ø
35.Ø21 SET YES =1
35.Ø22 SET NO =Ø
35.Ø3 DO PART 36 FOR X=(A+H):2*H:(B-H)
35.Ø4 SET T =2*T
35.Ø5 DO PART 36 FOR X=(A+2*H):2*H:(B-2*H)
35.Ø6 SET T=2*T
35.Ø7 DO PART 36 FOR X=A,B
35.Ø8 SET ARE=H*T/3
35.Ø85 TYPE ARE
```

```
35.Ø9Ø TYPE "DO YOU WISH ME TO REPEAT WITH A DIFFERENT VALUE OF N?"
35.Ø91 DEMAND ANS
35.Ø92 TO PART 34 IF ANS =1
35.Ø93 TYPE"DO YOU WISH ME TO REPEAT WITH DIFFERENT VALUES OF A AND B?"
35.Ø94 DEMAND ANS
35.Ø95 TO PART 33 IF ANS =1
35.Ø96 TYPE "DO YOU WISH ME TO REPEAT WITH A NEW FUNCTION?"
35.Ø97 DEMAND ANS
35.Ø98 TO PART 32 IF ANS =1
35.Ø99 DONE

36.Ø SET Y=X↑2+2*X+44
36.1 SET T=T+Y

37.1 TYPE "YOU GAVE ME A VALUE OF N THAT WAS NOT EVEN."
37.2 TO PART 34
```

Operation of Program:

```
DO PART 31

 I WILL DETERMINE BY SIMPSON'S RULE THE AREA
 UNDER THE CURVE Y=F(X) BETWEEN A AND B.  PLEASE NOTE
 THAT I WILL ASK YOU FOR THE VALUES OF A AND B AND ALSO FOR
 N, THE NUMBER OF SLICES YOU WANT ME TO MAKE.
 N MUST BE AN EVEN INTEGER.
YOU MUST TYPE THE COMMAND:  36.Ø SET Y=.....
 AND THEN TYPE: DO PART 33
←36.Ø SET Y=1
←DO PART 33
         A=Ø
         B=5
         N=4
       ARE=     5
DO YOU WISH ME TO REPEAT WITH A DIFFERENT VALUE OF N?
       ANS=YES
         N=5
YOU GAVE ME A VALUE OF N THAT WAS NOT EVEN.
         N=2
       ARE=     5
DO YOU WISH ME TO REPEAT WITH A DIFFERENT VALUE OF N?
       ANS=NO
DO YOU WISH ME TO REPEAT WITH DIFFERENT VALUES OF A AND B?
       ANS=NO
DO YOU WISH ME TO REPEAT WITH A NEW FUNCTION?
       ANS=YES
```

```
YOU MUST TYPE THE COMMAND:  36.Ø SET Y=.....
 AND THEN TYPE: DO PART 33
←36.Ø SET Y=X↑3+3X↑2-1.2X+2
←DO PART 33
         A=1
         B=7
         N=6
ERROR AT STEP 36                        (Note how the computer indicates)
LOOKS LIKE COMMA OR OPERATOR MISSING    (an error in the student's input)
←36.Ø SET Y=X↑3+3*X↑2-1.2*X+2           (of information in step 36.)
←DO PART 33
         A=2
         B=43.564
         N=8
       ARE=    9.82Ø438*1Ø↑5
DO YOU WISH ME TO REPEAT WITH A DIFFERENT VALUE OF N?
       ANS=YES
         N=5Ø
       ARE=    9.82Ø437*1Ø↑5
DO YOU WISH ME TO REPEAT WITH A DIFFERENT VALUE OF N?
       ANS=NO
DO YOU WISH ME TO REPEAT WITH DIFFERENT VALUES OF A AND B?
       ANS=NO
DO YOU WISH ME TO REPEAT WITH A NEW FUNCTION?
       ANS=NO
```

SIMTRP - Simpson's and Trapezoidal Rules

This program performs Simpson's and trapezoidal rule approximations to the integral of a single-valued function. The student specifies the function and the endpoints of the interval.

Listing of Program:

```
1 READ "YOUR NUMBER OF SUBINTERVALS IS ",N,#
1.01 TYPE "N MUST BE EVEN" IF FP(N/2)<>0
1.02 TO STEP 1 IF FP(N/2)<>0
1.1 TYPE "TYPE: DEFINE F(X)='YOUR EQUATION',ENTER,THEN TYPE GO"
1.2 STOP
1.3 READ "THE ENDPOINTS ON YOUR INTERVAL ARE ",A," AND ",B,#
1.31 TYPE "GIVE THE BIGGER ONE FIRST" IF A>B
1.32 TO STEP 1.3 IF A>B
1.4 X=A,T=0,S=0,Q=(B-A)/(2*N),R=Q*2/3,P=0
1.5 T=T+2*Q*'F(X)' IF X<>A&X<>B
1.6 T=T+Q*'F(X)',S=S+R*'F(X)' IF X=A@X=B
1.7 S=S+2*R*'F(X)' IF FP(P/2)=0&X<>A&X<>B
1.8 S=S+4*R*'F(X)' IF FP(P/2)<>0
1.9 X=X+2*Q,P=P+1
1.92 TO STEP 1.5 IF X<=B
1.93 PRINT "THE APPROXIMATE VALUE OF YOUR INTEGRAL FROM ",A,#
1.94 PRINT "TO ",B," USING ",N," SUBINTERVALS IS ",T,#
1.95 PRINT "BY THE TRAPEZOIDAL RULE AND ",S," BY SIMPSON'S RULE",#
```

Operation of Program:

```
←DO PART 1
YOUR NUMBER OF SUBINTERVALS IS 10
TYPE: DEFINE F(X)='YOUR EQUATION',ENTER,THEN TYPE GO
STOPPED AT STEP  1.2
←DEFINE F(X)=X↑2
←GO
THE ENDPOINTS ON YOUR INTERVAL ARE 0 AND 1
THE APPROXIMATE VALUE OF YOUR INTEGRAL FROM 0
TO 1 USING 10 SUBINTERVALS IS .335
BY THE TRAPEZOIDAL RULE AND .3333333 BY SIMPSON'S RULE
←
```

```
←DO PART 1

YOUR NUMBER OF SUBINTERVALS IS 10
TYPE: DEFINE F(X)='YOUR EQUATION',ENTER,THEN TYPE GO
STOPPED AT STEP  1.2
←DEFINE F(X)=X↑1/3
←GO
THE ENDPOINTS ON YOUR INTERVAL ARE 0 AND 1
THE APPROXIMATE VALUE OF YOUR INTEGRAL FROM 0
TO 1 USING 10 SUBINTERVALS IS .1666667
BY THE TRAPEZOIDAL RULE AND .1666667 BY SIMPSON'S RULE
←
```

APPROX - Approximation to a Derivative

The following program gives a reasonable approximation of the derivative of a continuous function. When looking for f'(P), the computer determines the slope of the line passing through points f(P+I) and f(P-I) where I is a positive constant. I is then made smaller by 1/10 and new slopes are calculated until the difference of consecutive slopes is less than the desired accuracy. Then the derivative is taken as the last slope. This program was written by a 12th grade student and reflects a complete understanding of the definition of a derivative.

Listing of Program:

```
1.1 SAY RETYPE STEP 2.2 WITH YOUR FUNCTION, IN THE INDEPENDENT
1.2 SAY VARIABLE X, IN THE FORM "2.2 SET Y(Z)=(FUNCTION IN X)",
1.3 SAY THEN DO PART 3

2.0 SET X=P-I IF Z=1
2.1 SET X=0 IF 'X'<.00001
2.2 SET Y(Z)=X
2.3 SET X=P+I

3.01 SET I=.01
3.02 SAY AT WHAT VALUE OF X DO YOU WANT THE DERIVATIVE?
3.03 DEMAND P
3.04 DO PART 2 FOR Z=1,2
3.05 SET DA=(Y(Z)-Y(Z-1))/(2*I)
3.06 SET DA=DB IF I<.01
3.07 SET I=I*.1
3.08 DO PART 2 FOR Z=1,2
3.09 SET DB=(Y(Z)-Y(Z-1))/(2*I)
3.10 TO STEP 3.12 IF 'DA-DB'<.01
3.11 TO STEP 3.06
3.12 SET DER=DB
3.13 SET DER=0 IF 'DER'<.001
3.14 SET DER=(IP((DER*10↑4+1)/10))/1000
3.15 SAY THE DERIVATIVE IS:
3.16 TYPE DER
```

Operation of Program:

```
RETYPE STEP 2.2 WITH YOUR FUNCTION, IN THE INDEPENDENT
VARIABLE X, IN THE FORM "2.2 SET Y(Z)=(FUNCTION IN X)",
THEN DO PART 3
*2.2 SET Y(Z)=X
*DO PART 3
AT WHAT VALUE OF X DO YOU WANT THE DERIVATIVE?
P=12
THE DERIVATIVE IS:
         DER =      1

*2.2 SET Y(Z)=SIN(X)
*DO PART 3
AT WHAT VALUE OF X DO YOU WANT THE DERIVATIVE?
P=0
THE DERIVATIVE IS:
         DER =      1

*DO PART 3
AT WHAT VALUE OF X DO YOU WANT THE DERIVATIVE?
P=3.14159536/2
THE DERIVATIVE IS:
         DER =      0

*2.2 SET Y(Z)=X↑2
*DO PART 3
AT WHAT VALUE OF X DO YOU WANT THE DERIVATIVE?
P=0
THE DERIVATIVE IS:
         DER =      0

*DO PART 3
AT WHAT VALUE OF X DO YOU WANT THE DERIVATIVE?
P=8
THE DERIVATIVE IS:
         DER =     16

*2.2 SET Y(Z)=3*X↑3+4*X↑2+1
*DO PART 3
AT WHAT VALUE OF X DO YOU WANT THE DERIVATIVE?
P=1
THE DERIVATIVE IS:
         DER =     17

*DO PART 3
AT WHAT VALUE OF X DO YOU WANT THE DERIVATIVE?
P=-10
THE DERIVATIVE IS:
         DER =      0
```

APPENDIX III

Section A: Independent Student Projects

Section B: Program Library Listings

APPENDIX III

Section A: Independent Student Projects

The following fifteen programs illustrate a wide range of applications developed by students in the course of a very large number of individual student projects. The wide range of topics shown, including biology, chemistry, physics, linguistics, mathematics, music, and games of strategy, reflect the diversity of student interests that was found in the schools participating in the Project.

1. Biology:	BIOLOG	- Statistical Biology
2. Chemistry:	CHEM	- Drill on Molecular Weights
	PPT	- Predicts Precipitates, Simulates Laboratory
	REACT	- Factors in Reactions
	TITRA	- Titration Laboratory
3. Physics:	IMAGES	- Mirror Optics Problem
4. Linguistics:	SNTNCS	- Generates Random Sentences.
	WORDS	- Generates Four-Letter Words
5. Mathematics:	DATE	- Finds Date for Day of Week
	ORDER	- Orders Numbers
6. Music:	MUSIC	- Generates a Random Melody
7. Games of Strategy:	CANMIS	- Cannibal-Missionary Game
	NAVIG	- Exercise in Navigation
	007	- Chemin de Fer
	PLANES	- Exercise in Probability

BIOLOG - Statistical Biology

This program was written to analyze statistical information gathered in an experimental project concerned with weight changes in mice caused by variations in food intake. It was used to summarize data gathered daily by the student - "Pam" - over a period of several weeks. The output produces mean, standard deviation, and error for each set of data.

Listing of Program:

```
1.00 TYPE "THIS IS FOR A PROJECT I WAS WORKING ON-PAM"
1.001 A=1,Q=1
1.01 TYPE "I CAN FIND VALUE, INCREASE IN GROWTH OF 2 TEST GROUPS"
1.02 TYPE "ALSO, FIND VALUE OF ERROR"
1.03 READ "HOW MANY WEIGHT READINGS OF MICE? ",I,#
1.04 TYPE "I ASK FIRST FOR BACKGROUND AND WT COUNT FOR NORM MICE"
1.05 TYPE "SECOND, BACKGROUND AND WT FOR TEST MICE"
1.06 DEMAND BN[X] FOR X=1:1:I
1.07 DEMAND N[X] FOR X=1:1:I
1.08 DEMAND BT[X] FOR X=1:1:I
1.09 DEMAND T[X] FOR X=1:1:I
1.1 SBN=BN[A]+BN[A+1],SN=N[A]+N[A+1],SBT=BT[A]+BT[A+1]
1.11 ST=T[A]+T[A+1],A=A+2
1.12 SBN=SBN+BN[A],SN=SN+N[A],SBT=SBT+BT[A],ST=ST+T[A],A=A+1
1.13 TO STEP 1.12 IF A<=I
1.14 SBN=SBN/I,SN=SN/I,SBT=SBT/I,ST=ST/I
1.145 TYPE "THESE ARE THE MEANS IN ORDER ",SBN,SN,SBT,ST
1.15 BN[H]=BN[H]-SBN,N[H]=N[H]-SN,BT[H]=BT[H]-SBT FOR H=1:1:I
1.16 T[H]=T[H]-ST FOR H=1:1:I
1.17 VBN=BN[Q]↑2+BN[Q+1]↑2,VN=N[Q]↑2+N[Q+1]↑2
1.18 VBT=BT[Q]↑2+BT[Q+1]↑2,VT=T[Q]↑2+T[Q+1]↑2,Q=Q+2
1.19 VBN=VBN+BN[Q]↑2,VN=VN+N[Q]↑2,VBT=VBT+BT[Q]↑2,VT=VT+T[Q]↑2
1.191 Q=Q+1
1.2 TO STEP 1.19 IF Q<=I
1.21 VBN=VBN/(I-1),VN=VN/(I-1),VBT=VBT/(I-1),VT=VT/(I-1)
1.215 VBS=SQRT(VBN),VS=SQRT(VN),VBR=SQRT(VBT),VR=SQRT(VT)
1.216 TYPE "THESE ARE THE STANDARD DEVIATIONS IN ORDER",VBS,VS,VBR
1.217 TYPE VR
1.22 NCN=SN-SBN,NCT=ST-SBT,EN=SQRT(VBN+VN),ET=SQRT(VBT+VT)
1.225 TYPE "THESE ARE THE NET COUNTS IN ORDER",NCN,NCT
1.23 INC=NCT-NCN,ERR=SQRT((VBN+VN)+(VBT+VT))
1.24 PRINT "VALUE OF INCREASE IN GROWTH = ",INC,#
1.25 PRINT "VALUE OF ERROR = ",ERR,#
1.255 TYPE "PLEASE DELETE ALL VALUES BEFORE STARTING AGAIN"
```

Output of Program:

```
THIS IS FOR A PROJECT I WAS WORKING ON-PAM
I CAN FIND VALUE INCREASE IN GROWTH OF 2 TEST GROUPS
ALSO, FIND VALUE OF ERROR
HOW MANY WEIGHT READINGS OF MICE? 5
I ASK FIRST FOR BACKGROUND AND WT COUNT FOR NORM MICE
SECOND, BACKGROUND AND WT FOR TEST MICE
       BN[1]=6
       BN[2]=8
       BN[3]=7
       BN[4]=9
       BN[5]=10
        N[1]=12
        N[2]=13
        N[3]=12
        N[4]=11
        N[5]=10
       BT[1]=8
       BT[2]=10
       BT[3]=9
       BT[4]=13
       BT[5]=15
        T[1]=18
        T[2]=17
        T[3]=17
        T[4]=19
        T[5]=14
THESE ARE THE MEANS IN ORDER
         SBN=     8
          SN=    11.6
         SBT=    11
          ST=    17
THESE ARE THE STANDARD DEVIATIONS IN ORDER
         VBS=     1.581139
          VS=     1.140175
         VBR=     2.915476
          VR=     1.870829
THESE ARE THE NET COUNTS IN ORDER
         NCN=     3.6
         NCT=     6
VALUE OF INCREASE IN GROWTH = 2.4
VALUE OF ERROR = 3.974921
PLEASE DELETE ALL VALUES BEFORE STARTING AGAIN
```

CHEM - Drill Exercise on Molecular Weights

This program demonstrates the use of TELCOMP as a device to introduce new material, and to execute rote work and drill. The subject which it purports to teach is the calculation of molecular weights; a conceptually easy topic which every student encounters in his first year of chemistry.

It begins by explaining the method the student must use in calculating molecular weights, and gives an example. It then asks the student to do a trivial one-step problem. Should he do it wrongly, it will explain again how to solve it. If he is right, it will proceed directly to working him through a more complex example, compounded from steps like the one he has done. Once he has successfully completed this, the program poses him a series of "homework" or drill problems, offering only "right" or "wrong" diagnostics. When the student has finished this problem set, the program is finished.

Listing of Program:

```
1.01 YES=1,NO=2
1.06 L[1]=18,L[2]=44
1.07 L[3]=40,L[4]=98,L[5]=114
1.08 L[6]=142,L[7]=84
1.09 H=1,C=12,NA=23,S=32,O=16,CL=35.5,N=0
1.2 DO PART 21

2.5 TO STEP21.40

21.01 TYPE #,"FINDING THE MOLECULAR WEIGHT OF A COMPOUND.",#
21.02 TYPE "TODAY YOU ARE GOING TO BE TAUGHT A VERY EASY BUT"
21.04 TYPE "IMPORTANT PART OF CHEMISTRY.",#
21.06 TYPE "THE MOLECULAR WEGHT OF AN ELEMENT IS THE NUMBER"
21.07 TYPE "OF GRAMS IN A MOLE.  FOR EXAMPLE, THE ATOMIC"
21.08 TYPE "WEIGHT OF OXYGEN IS 16 -- ONE MOLE OF OXYGEN ATOMS"
21.09 TYPE "(6.02 X 10↑23) WEIGHS 16 GRAMS.  ONE MOLE"
21.10 TYPE "OF O(2) WOULD HAVE THE MOLECULAR WEIGHT OF 16 X 2 = 32."
21.11 TYPE #,"FOR EXAMPLE, HOW MANY GRAMS ARE THERE IN 2 MOLES"
21.13 READ "OF TIN? (TIN HAS THE ATOMIC WEIGHT OF 118.7) ",ANS,#,#
21.16 TO STEP 21.18 IF ANS=237.4
21.17 TYPE "NO, THE ANSWER IS 2 X 118.7 = 237.4"
21.175 TO STEP 21.19
21.18 TYPE "RIGHT"
21.19 TYPE #,"NOW TO FIND THE GRAMS IN A MOLE OF THE COMPOUND NH(3)"
21.20 TYPE #,"THERE IS ONE MOLE OF NITROGEN IN THE COMPOUND WHICH"
21.21 TYPE "WEIGHS 14 GRAMS.  THERE ARE 3 MOLES OF HYDROGEN ATOMS"
21.22 TYPE "WHICH WEIGH 1 GRAM EACH.  THE SUM TOTAL EQUALS THE"
21.23 TYPE "MOLECULAR WEIGHT --- 14 + 3*1 =17",#
21.24 TYPE "YOU WILL NEED TO KNOW THE FOLLOWING ATOMIC WEIGHTS:"
21.25 PRINT "      H =  1.0",#,"      C = 12.0",#,"     O = 16.0",#
21.26 PRINT "     NA = 23.0",#,"      S = 32.0",#,"    CL = 35.5",#
21.31 TYPE #,"LET'S DO C H CL(3) TOGETHER.",#
21.32 READ "THE NUMBER OF MOLES OF CARBON TIMES ITS AT. WT. IS? ",ANS
21.33 PRINT #,#
21.34 DO STEP 22.1 IF ANS><12.0
21.35 READ #,"THE NUMBER OF MOLES OF H TIMES ITS AT.WT. IS? ",ANS,#,#
21.37 DO STEP 22.2 IF ANS><1
21.38 READ #,"THE NUMBER OF MOLES OF CL TIMES ITS AT.WT. IS? ",ANS
21.39 PRINT #,#
21.395 DO STEP 22.3 IF ANS><106.5
21.40 READ #,"THE SUM OF THESE MOLECULAR WEIGHTS IS? ",ANS,#,#
21.41 TO STEP 22.4 IF ANS><119.5
21.43 TYPE #,"VERY GOOD.  NOW LET'S SEE IF YOU CAN DO"
```

```
21.44 TYPE "SOME BY YOURSELF.",#
21.51 PRINT "    1) H(2)O",#,"    2) CO(2)",#,"    3) NA OH",#
21.52 PRINT "    4) H(2) SO(4)",#,"    5) C(8)H(18)",#
21.53 PRINT "    6) NA(2)SO(4)",#,"    7) NA H CO(3)",#
21.677 READ #,"WHICH ONE ARE YOU GOING TO DO NOW? ",PRB
21.69 READ #,#,"WHAT IS THE MOLECULAR WEIGHT OF THAT COMPOUND? ",ANS
21.70 READ #,#
21.80 TO PART 23 IF L[PRB]=ANS
21.81 READ "WRONG. DO YOU STILL NEED HELP? ",ANS,#
21.82 PRINT "THEN TRY ONCE MORE. WATCH YOUR ARITHMETIC. ",# IF ANS=2
21.83 TO PART 21.69 IF ANS=2
21.84 READ "WHAT IS THE FIRST ELEMENT IN THE COMPOUND? ",ELE,#
21.845 READ "HOW MANY MOLES OF THAT ELEMENT ARE THERE? ",MOLE,#
21.85 PRINT "THEREFORE, THE WEIGHT OF "
21.86 DO PART ELE+30
21.87 PRINT" IN THE COMPOUND IS ",MOLE," * ",ELE," = ",MOLE*ELE,#,#
21.88 READ "IS THERE ANOTHER ELEMENT IN THE COMPOUND? ",ANS,#
21.89 TO STEP 21.91 IF ANS=NO
21.895 READ "WHAT IS IT? ",ELE,#
21.90 TO STEP 21.845
21.91 PRINT "THE SUM OF THESE WEIGHTS EQUAL THE MOLECULAR",#
21.912 PRINT "WEIGHT OF THE COMPOUND.",#
21.92 TO STEP 21.69

22.1 TYPE "THE AT.WT. OF C IS 12.   12*1=12",#
22.2 TYPE "THE AT.WT OF H IS 1.   1*1=1",#
22.3 TYPE "THE AT. WT. OF CL IS 35.5.   35.5*3=106.5",#
22.4 TYPE "YOU HAVE ARITHMETIC PROBLEMS.  TRY AGAIN.",#
22.5 TO STEP 21.40

23.01 READ #,"VERY GOOD.  LIKE TO TRY ANOTHER ONE? ",ANS,#,#
23.5 TO PART 21.67 IF ANS=1
23.6 DONE

31 PRINT "H"

42 PRINT "C"

46 PRINT "O"

53 PRINT "NA"

62 PRINT "S"

65.5 PRINT "CL"
```

Operation of Program:

FINDING THE MOLECULAR WEIGHT OF A COMPOUND.

TODAY YOU ARE GOING TO BE TAUGHT A VERY EASY BUT IMPORTANT PART OF CHEMISTRY.

THE MOLECULAR WEGHT OF AN ELEMENT IS THE NUMBER OF GRAMS IN A MOLE. FOR EXAMPLE, THE ATOMIC WEIGHT OF OXYGEN IS 16 -- ONE MOLE OF OXYGEN ATOMS (6.02 X 10↑23) WEIGHS 16 GRAMS. ONE MOLE OF O(2) WOULD HAVE THE MOLECULAR WEIGHT OF 16 X 2 = 32.

FOR EXAMPLE, HOW MANY GRAMS ARE THERE IN 2 MOLES OF TIN? (TIN HAS THE ATOMIC WEIGHT OF 118.7) 237.4

RIGHT

NOW TO FIND THE GRAMS IN A MOLE OF THE COMPOUND NH(3)

THERE IS ONE MOLE OF NITROGEN IN THE COMPOUND WHICH WEIGHS 14 GRAMS. THERE ARE 3 MOLES OF HYDROGEN ATOMS WHICH WEIGH 1 GRAM EACH. THE SUM TOTAL EQUALS THE MOLECULAR --- 14 + 3*1 =17

YOU WILL NEED TO KNOW THE FOLLOWING ATOMIC WEIGHTS:

```
 H =  1.0
 C = 12.0
 O = 16.0
NA = 23.0
 S = 32.0
CL = 35.5
```

LET'S DO C H CL(3) TOGETHER.

THE NUMBER OF MOLES OF CARBON TIMES ITS AT. WT. IS? 14

THE AT.WT. OF C IS 12. 12*1=12

THE NUMBER OF MOLES OF H TIMES ITS AT.WT. IS? 1

THE NUMBER OF MOLES OF CL TIMES ITS AT.WT. IS? 106.5

THE SUM OF THESE MOLECULAR WEIGHTS IS? 116.5

YOU HAVE ARITHMETIC PROBLEMS. TRY AGAIN.

THE SUM OF THESE MOLECULAR WEIGHTS IS? 119.6\5

VERY GOOD. NOW LET'S SEE IF YOU CAN DO
SOME BY YOURSELF.

1) H(2)O
2) CO(2)
3) NA OH
4) H(2) SO(4)
5) C(8)H(18)
6) NA(2)SO(4)
7) NA H CO(3)

WHICH ONE ARE YOU GOING TO DO NOW? 1

WHAT IS THE MOLECULAR WEIGHT OF THAT COMPOUND? 18

VERY GOOD. LIKE TO TRY ANOTHER ONE? YES

WHICH ONE ARE YOU GOING TO DO NOW? 2

WHAT IS THE MOLECULAR WEIGHT OF THAT COMPOUND? 441

WRONG. DO YOU STILL NEED HELP? NO
THEN TRY ONCE MORE. WATCH YOUR ARITHMETIC.

WHAT IS THE MOLECULAR WEIGHT OF THAT COMPOUND? 44

VERY GOOD. LIKE TO TRY ANOTHER ONE? YES

```
WHICH ONE ARE YOU GOING TO DO NOW? 3

WHAT IS THE MOLECULAR WEIGHT OF THAT COMPOUND? 123

WRONG. DO YOU STILL NEED HELP? YES
WHAT IS THE FIRST ELEMENT IN THE COMPOUND? NA
HOW MANY MOLES OF THAT ELEMENT ARE THERE? 1
THEREFORE, THE WEIGHT OF NA IN THE COMPOUND IS 1 * 23 = 23

IS THERE ANOTHER ELEMENT IN THE COMPOUND? YES
WHAT IS IT? O
HOW MANY MOLES OF THAT ELEMENT ARE THERE? 1
THEREFORE, THE WEIGHT OF O IN THE COMPOUND IS 1 * 16 = 16

IS THERE ANOTHER ELEMENT IN THE COMPOUND? YES
WHAT IS IT? H
HOW MANY MOLES OF THAT ELEMENT ARE THERE? 1
THEREFORE, THE WEIGHT OF H IN THE COMPOUND IS 1 * 1 = 1

IS THERE ANOTHER ELEMENT IN THE COMPOUND? NO
THE SUM OF THESE WEIGHTS EQUAL THE MOLECULAR
WEIGHT OF THE COMPOUND.


WHAT IS THE MOLECULAR WEIGHT OF THAT COMPOUND? 40


VERY GOOD.  LIKE TO TRY ANOTHER ONE? NO
```

PPT - Laboratory Simulation

PPT demonstrates in a very primitive manner one of the little explored but potentially very useful functions of the computer. It simulates a laboratory. The student using it feeds in the anions and the cations of two solutions which he is reacting, and the program will tell him what, if any, precipitates he might expect.

It is not implied that all the benefits a student gains from working in a laboratory will accrue to him from running this program. However, in certain types of experiment where the work is repetitive and the data massive and unwieldy, a computer simulating an experiment that the student has sampled in the laboratory, might be economical and stimulating.

In any case, the student learns a great deal by writing such a laboratory program.

Listing of Program:

```
1.02 TYPE #,"YOU HAVE TWO SOLUTION, EACH WITH"
1.03 TYPE "ONE CATION AND ONE ANION.  I HAVE"
1.04 TYPE "ASSIGNED EACH ION A NUMBER."
1.05 TYPE "I WILL ASK FOR THE NUMBERS OF THE "
1.06 TYPE"IONS IN THE FIRST SOLUTION [C(1) AND A(1)]"
1.07 TYPE "AND THOSE FOR THE SECOND SOLUTION"
1.08 TYPE "TYPE [C(2) AND A(2)], AND WILL TELL YOU WHICH,"
1.09 TYPE "IF ANY, PRECIPITATES YOU MAY EXPECT.",#
1.10 TYPE "     CATIONS:"
1.11 PRINT "H+ =2",#,"LI+ =3",#,"NH(4)+, NA+, K+ =5",#
1.12 PRINT "AG+ =7",#,"PB++ =11",#,"HG(2)++ =13",#
1.13 PRINT "BA++, SR++ =17",#,"BE++, MG++, CA++ =19",#
1.14 TYPE #,"OTHER CATIONS =23",#
1.15 TYPE "     ANIONS:"
1.16 PRINT "NO(3)- =1",#,"CH(3)COO- =29",#
1.17 PRINT "CL-, BR-, I-,  =31",#,"SO(4)--, CRO(4)--  =37",#
1.18 PRINT "S-- =41",#,"OH- =43",#,"CO(3)--, PO(4)-- =47",#,#
1.315 TYPE "CON =C(1), CTW = C(2), AON = A(1), ATW = A82)"
1.32 DEMAND CON, AON, CTW, ATW
1.36 F=CON*ATW, G=CTW*AON
1.38 TO STEP 1.54 IF CON=CTW OR AON=ATW
```

```
1.43 A=10↑-2
1.44 DO STEP F*A IF F=203 @ F=217 @ F=342 OR F=403 @ F=629
1.45 DO STEP F*A IF F=407@F=259@F=287@F=454@F=533
1.46 DO STEP F*A IF F=943@F=301@F=473@F=559@F=817
1.47 DO STEP F*A IF F=989@F=329@F=517@F=611
1.48 DO STEP F*A IF F=799@F=893@F=1081
1.481 DO STEP 2.6 IF F=82
1.482 DO STEP 3.3 IF F=141
1.49TO STEP 1.52 IF F=G
1.50 F=G
1.51 TO STEP 1.44
1.52 TYPE "NO OTHER PRECIPITATES",#,#
1.53 TO STEP 1.32
1.54 TYPE "NO PRECIPITATES AT ALL.",#
1.55 TO STEP 1.32

2.03 TYPE"AG CH(3)COO"
2.17 TYPE"AG CL, BR, I"
2.59 TYPE"AG SO(4), CTO(4)"
2.6 TYPE " H S "
2.87 TYPE "AG S"

3.01 TYPE "AG OH"
3.29 TYPE "AG CO(3), PO(4)"
3.3 TYPE "LI CO(3), PO(4)"
3.43 TYPE "PB CL, BR, I"

4.03 TYPE "HG(2) CL, BR, I"
4.07 TYPE "PB SO(4), CRO(4)"
4.54 TYPE "PB S"
4.73 TYPE"PB OH "

5.17 TYPE"PB CO(3), PO(4)"
5.33 TYPE "HG(2) S"
5.59 TYPE "HG(2) OH"

6.11 TYPE "HG(2) CO(3), PO(4)"
6.29 TYPE "BA, SR CO(4), CRO(4)"

7.99 TYPE "BA, SR CO(3), PO(4)"

8.17 TYPE "BE, MG, CA OH"
8.93 TYPE "BE, MG, CA CO(3), PO(4)"

9.43 TYPE "OTHER SULFIDE"
9.89 TYPE "OTHER HYDROXIDE"

10.81 TYPE"OTHER CARBONATE OFPHOSPHATE"

←
```

Output of Program:

```
YOU HAVE TWO SOLUTIONS,EACH WITH
ONE CATION AND ONE ANION.  I HAVE
ASSIGNED EACH ION A NUMBER.
 I WILL ASK FOR THE NUMBERS OF THE
IONS IN THE FIRST SOLUTION [C(1) AND A(1)]
AND THOSE FOR THE SECOND SOLUTION
[C(2) AND A(2)],AND WILL TELL YOU WHICH, IF ANY,
PRECIPITATES YOU MAY EXPECT.

      CATIONS:
H+ =2
LI+  =3
NH(4)+, NA+, K+  =5
AG+  =7
PB++  =11
HG(2)++ =13
BA++,  SR++  =17
BE++, MG++, CA++  =19
OTHER CATIONS = 23

      ANIONS:
NO(3)-  =1
CH(3)COO-  =29
CL-, BR-, I-,  =31
SO(4)--, CRO(4)--  =37
S--  =41
OH-  =43
CO(3)--, PO(4)-- =47

CON=C(1), CTW = C(2), AON = A(1), ATW = A(2)
       CON=13
       AON=41
       CTW=7
       ATW=43
HG (2) OH
AG S
NO OTHER PRECIPITATES.
       CON=17
       AON=41
       CTW=11
       ATW=47
BA, SR CO(3), PO(4)
NO OTHER PRECIPITATES.
       CON=2
       AON=43
       CTW=5
       ATW=31
NO OTHER PRECIPITATES.
       CON=23
       AON=37
       CTW=23
       ATW=47
NO PRECIPITATES AT ALL.
```

REACT—A Chemistry Practice Program

One of the high school science programs is a program that computes the correct mole amounts for a simple two-element reaction, either covalent or ionic. Because of the irregularities of chemistry, the program is correct only about 99% of the time. Although the completed program is a useful one, this is an excellent program to assign to students to write, for it necessitates a thorough analysis of the periodic table.

Listing of Program:

```
1.0001 LINE
1.001 SAY GIVE ME THE ATOMIC NUMBERS OF TWO ELEMENTS ON THE
1.002 SAY PERIODIC TABLE, BELOW NUMBER 55, AND I WILL TRY
1.003 SAY TO GIVE PROPORTIONS FOR ONE OR TWO POSSIBLE COMPOUNDS.
1.05 LINE
1.1 DEMAND X(1)
1.2 DEMAND X(2)
1.21 TO STEP 8.1 IF X(1)=8 AND X(2)=8
1.22 TO STEP 8.1 IF X(1)=9 AND X(2)=9
1.23 TO STEP 8.1 IF X(1)=17 AND X(2)=17
1.24 TO STEP 8.1 IF X(1)=35 AND X(2)=35
1.25 TO STEP 8.1 IF X(1)=53 AND X(2)=53
1.3 DO PART 2 FOR I=1,2
1.34 TO STEP 5.1 IF V(1)=-1 OR V(2)=-1
1.4 TO STEP 3.1 IF (V(1)-4)*(V(2)-4)>.5
1.405 TO STEP 4.1 IF V(1)<.5 OR V(2)<.5
1.41 TO STEP 3.1 IF V(1)=4 AND V(2)=4
1.42 DO PART 10 FOR A=1(1)3
1.425 TO STEP 6.1 IF X(1)=1 AND V(1)=1
1.426 TO STEP 7.1 IF X(2)=1 AND V(2)=1
1.43 TO STEP 5.1 IF V(1)=-1 OR V(2)=-1

2.01 TO STEP 2.1 IF X(I)>1.5
2.02 SET V(I)=1
2.03 DONE
2.1 TO STEP 2.2 IF X(I)<25.5 OR X(I)>28.5
2.11 SET V(I)=2
2.12 DONE
```

```
2.2 TO STEP 2.3 IF X(I)<43.5 OR X(I)>46.5
2.21 SET V(I)=4
2.22 DONE
2.3 TO STEP 2.4 IF X(I)>25.5
2.31 SET V(I)=IP(8*(FP((X(I)-1.99)/8)))
2.32 DONE
2.4 TO STEP 2.5 IF X(I)>43.5
2.41 SET V(I)=IP(8*FP((X(I)-27.99)/8)))
2.42 DONE
2.5 TO STEP 2.6 IF X(I)>54.5 OR X(I)<46.5
2.51 SET V(I)=IP(8*FP((X(I)-45.99)/8))
2.52 DONE

3.05 TO STEP 1.425 IF X(1)=1 OR X(2)=1
3.1 LINE
3.15 TO STEP 1.425 IF X(1)=1 OR X(2)=1
3.2 SAY THAT IS THE BEST I CAN DO

4.1 SAY ONE OF THESE ELEMENTS IS COMMONLY CONSIDERED INERT.

5.1 SAY ONE OF THESE ELEMENTS HAS TOO LARGE AN ATOMIC NUMBER

6.1 SET V(1)=7
6.2 TO STEP 1.34

7.05 DONE IF V(1)=7
7.1 SET V(2)=7
7.2 TO STEP 1.34

8.1 SET A=1
8.2 SET B=1
8.3 TO STEP 11.3

10.1 DO PART 11 FOR B=1(1)3

11.1 DONE IF FP((A*V(1)+B*V(2))/8)>.01
11.2 DONE IF A=B AND 'A-1'>.1
11.3 SAY ONE COMPOUND USES A MOLES OF X(1) AND B MOLES OF X(2)
11.4 TYPE A,B
11.5 LINE
```

Operation of Program:

```
GIVE ME THE ATOMIC NUMBERS OF TWO ELEMENTS ON THE
PERIODIC TABLE, BELOW NUMBER 55, AND I WILL TRY
TO GIVE PROPORTIONS FOR ONE OR TWO POSSIBLE COMPOUNDS.

X(1)=1
X(2)=1

ONE COMPOUND USES A MOLES OF X(1) AND B MOLES OF X(2)
              A =      1
              B =      1



GIVE ME THE ATOMIC NUMBERS OF TWO ELEMENTS ON THE
PERIODIC TABLE, BELOW NUMBER 55, AND I WILL TRY
TO GIVE PROPORTIONS FOR ONE OR TWO POSSIBLE COMPOUNDS.

X(1)=1
X(2)=8
ONE COMPOUND USES A MOLES OF X(1) AND B MOLES OF X(2)
              A =      2
              B =      1



GIVE ME THE ATOMIC NUMBERS OF TWO ELEMENTS ON THE
PERIODIC TABLE, BELOW NUMBER 55, AND I WILL TRY
TO GIVE PROPORTIONS FOR ONE OR TWO POSSIBLE COMPOUNDS.

X(1)=2
X(2)=7
ONE OF THESE ELEMENTS IS COMMONLY CONSIDERED INERT.


GIVE ME THE ATOMIC NUMBERS OF TWO ELEMENTS ON THE
PERIODIC TABLE, BELOW NUMBER 55, AND I WILL TRY
TO GIVE PROPORTIONS FOR ONE OR TWO POSSIBLE COMPOUNDS.

X(1)=11
X(2)=17
ONE COMPOUND USES A MOLES OF X(1) AND B MOLES OF X(2)
              A =      1
              B =      1
```

TITRA - Titration Lab Program

This program was written by a Lexington High School chemistry student. It is designed to guide a student step by step through a titration laboratory. The program is an example of the use of a computer as an aid in a specific subject area. A complete understanding of the laboratory problem was, of course, needed in order to write the program. It then serves as a time-saver by doing the necessary calculations.

Listing of Program:

```
1.01 SAY  THE PURPOSE OF THIS LAB IS TO DISCOVER THE
1.02 SAY CONCENTRATION OF A SOLUTION BY TITRATION.
1.03 LINE
1.04 SAY LET A=THE ML OF ACID HCL USED
1.05 DEMAND A
1.06 LINE
1.07 SAY NOW LET B=THE CONCENTRATION OF THE ACID
1.08 DEMAND B
1.09 LINE
1.10 SET C=A*10↑-3*B
1.11 SAY THE AMOUNT OF HCL USED IS C AND
1.12 TYPE C
1.13 LINE
1.14 SAY SINCE HCL+NAOH GIVES NACL+H(2)O, C MOLES OF
1.15 SAY NAOH ARE USED.
1.16 LINE
1.17 SAY LET D=THE AMOUNT IN ML OF THE BASE NEEDED IN TITRATION.
1.18 LINE
1.20 DEMAND D
1.21 LINE
1.22 SET E=C/(D*10↑-3)
1.23 SAY THE MOLARITY OF THE BASE IS E AND
1.24 TYPE E
1.25 LINE
1.26 SAY NOW FOR PART II OF THE LAB. THE BASE ABOVE WITH
1.27 SAY MOLARITY E IS USED TO DETERMINE THE GRAMS OF ACID/MOLE
1.28 SAY OF BASE.
1.29 LINE
```

```
1.30 SAY LET F=THE AMOUNT IN ML OF BASE USED.
1.32 DEMAND F
1.33 LINE
1.34 SET G=F*10↑-3
1.35 SAY THE NUMBER OF MOLES OF BASE USED IS G, AND
1.36 TYPE G
1.37 LINE
1.38 SAY NOW SAY H=THE GRAMS OF ACID USED.
1.39 DEMAND H
1.40 LINE
1.41 SET I=H/G
1.42 SAY THE NUMBER OF GRAMS OF ACID/MOLE IS I AND
1.43 TYPE I
1.44 LINE
1.45 SAY WHAT IS THE TRUE WEIGHT OF THE ACID.
1.46 DEMAND J
1.47 LINE
1.48 SET K=[(J-I)/J]*100
1.49 SAY THE PERCENTAGE OF ERROR IS
1.50 TYPE K
1.51 LINE
```

Operation of Program:

```
THE PURPOSE OF THIS LAB IS TO DISCOVER THF
CONCENTRATION OF A SOLUTION BY TITRATION.

LET A=THE ML OF ACID HCL USED
A=24

NOW LET B=THE CONCENTRATION OF THE ACID
B=2

SAY THE AMOUNT OF HCL USED IS C AND
          C =        .048


SINCE HCL+NAOH GIVES NACL +H(2)O, C MOLES OF
NAOH ARE USED.

LET D=THE AMOUNT IN ML OF THE BASE NEEDED IN TITRATION.

D=18

THE MOLARITY OF THE BASE IS E AND
          E =      2.666667


NOW FOR PART II OF THE LAB. THE BASE ABOVE WITH
MOLARITY E IS USED TO DETERMINE THE GRAMS OF ACID/MOLE
OF BASE.

LET F=THE AMOUNT IN ML OF BASE USED.
F=50

THE NUMBER OF MOLES OF BASE USED IS G, AND
          G =        .05

NOW SAY H=THE GRAMS OF ACID USED.
H=11

THE NUMBER OF GRAMS OF ACID/MOLE IS I, AND
          I =     220

WHAT IS THE TRUE WEIGHT OF THE ACID.
J=215

THE PERCENTAGE OF ERROR IS
          K =     -2.32558.
```

IMAGES - Mirror Optics Problem

This program determines the relationship between the image, object, and focal point of a concave mirror. The student can enter values for any four of five variables; the value of the fifth, or unknown, variable will be computed and printed out. In addition, the characteristics of the image are listed for each set of values given.

Listing of Program:

```
1.00 TYPE "THIS PROGRAM FINDS THE MISSING INFORMATION"
1.001 TYPE "NEEDED TO DETERMINE THE RELATIONSHIP BETWEEN"
1.002 TYPE "THE IMAGE,OBJECT, AND FOCAL POINT OF A CONCAVE"
1.003 TYPE "MIRROR. ",#
1.0031 TYPE " F=FOCAL LENGTH OF MIRROR"
1.0032 TYPE "HO=HEIGHT OF OBJECT"
1.0033 TYPE "SO=DISTANCE OF OBJECT FROM FOCAL POINT"
1.0034 TYPE "SI=DISTANCE OF IMAGE FROM FOCAL POINT"
1.0035 TYPE "HI=HEIGHT OF IMAGE",#
1.004 TYPE "SUPPLY ALL VALUES THAT YOU HAVE - TYPE NONE IF"
1.005 TYPE "YOU DO NOT KNOW THE VALUE REQUESTED."
1.006 NONE=0,NO=0,YES=1,ANS=3
1.007 DEMAND F,HO,SO,SI,HI
1.008 TYPE "GIVE ME A CHALLENGE",# IF F>0&HI>0&HO>0&SI>0&SO>0
1.009 TO STEP 1.06 IF F>0&HI>0&HO>0&SI>0&SO>0
1.01 F=SQRT(SI*SO) IF F=0&SI><0&SO><0
1.011 HO=(HI*F)/SI IF HO=0&HI><0&F>0&SI><0
1.012 SO=F↑2/SI IF SO=0&F>0&SI><0
1.013 SI=F↑2/SO IF SI=0&F>0&SO><0
1.014 HI=(HO*SI)/F IF HI=0&HO><0&SI><0&F>0
1.015 HO=(HI*F)/SI IF HO=0&F>0&HI>0&SI><0
1.016 TYPE "INSUFFICIENT DATA" IF SO=0&F=0@SI=0&F=0@SI=0&SO=0
1.0165 TYPE "INSUFFICIENT DATA" IF HI=0&HO=0
1.017 TO STEP 1.06 IF SO=0&F=0@SI=0&F=0@SI=0&SO=0@HI=0&HO=0
1.018 READ "IS OBJECT FACING CONCAVE SIDE? ",POS,#
1.019 READ "IS OBJECT ON MIRROR SIDE OF F? ",ANS,# IF SO<F
1.02 TYPE "THE COMPLETE SET OF VALUES IS:"
1.021 TYPE F,HO,SO,SI,HI
1.03 TYPE "IMAGE IS REAL AND INVERTED.",# IF SO>F&POS=1
1.035 TYPE "IMAGE IS REAL AND INVERTED.",# IF SO<F&ANS=0&POS=1
1.04 TYPE "IMAGE IS VIRTUAL,BEHIND MIRROR,ERECT",# IF SO<F&ANS=1&POS=1
1.05 TYPE "IMAGE IS VIRTUAL,ON CONCAVE SIDE,ERECT.",# IF POS=0
1.06 TYPE #,#
1.07 TO STEP 1.007
```

Output of Program:

```
THIS PROGRAM FINDS THE MISSING INFORMATION
NEEDED TO DETERMINE THE RELATIONSHIP BETWEEN
THE IMAGE,OBJECT, AND FOCAL POINT OF A CONCAVE
MIRROR.

 F=FOCAL LENGTH OF MIRROR
HO=HEIGHT OF OBJECT
SO=DISTANCE OF OBJECT FROM FOCAL POINT
SI=DISTANCE OF IMAGE FROM FOCAL POINT
HI=HEIGHT OF IMAGE

SUPPLY ALL VALUES THAT YOU HAVE - TYPE NONE IF
YOU DO NOT KNOW THE VALUE REQUESTED.
           F=6
          HO=2
          SO=1
          SI=NONE
          HI=NONE
IS OBJECT FACING CONCAVE SIDE? NO
IS OBJECT ON MIRROR SIDE OF F? YES
THE COMPLETE SET OF VALUES IS:
           F=     6
          HO=     2
          SO=     1
          SI=    36
          HI=    12
IMAGE IS VIRTUAL,ON CONCAVE SIDE,ERECT.


           F=6
          HO=2
          SO=1
          SI=NONE
          HI=NONE
IS OBJECT FACING CONCAVE SIDE? YES
IS OBJECT ON MIRROR SIDE OF F? YES
THE COMPLETE SET OF VALUES IS:
           F=     6
          HO=     2
          SO=     1
          SI=    36
          HI=    12
IMAGE IS VIRTUAL,BEHIND MIRROR,ERECT
```

SNTNCS - A Program That Constructs Random English Sentences From a Set of Production Rules (Rules of Grammatic Structure) and a Dictionary

This impressive program exemplifies another non-numerical mathematical application, and makes a minor contribution to structural linguistics. It is an exercise in the mathematics of sentence structure written by Larry Rich, a 12th grade student at Lexington High School. One of the English teachers there used it to aid low level students of senior English to perceive sentence structure and errors in use of words. He wants to use programs of this type more extensively next year. The program has been a great popular success with the sixth graders in Winchester, in the high schools, and also with some of the programmers at BBN. The second transcript of the program operation was done with a revised dictionary prepared, obviously, by some programmers.

Listing of Program:

```
1.01 DO PART 2
1.02 SET A=IP(RAN(5))+6
1.03 DO PART A
1.035 SET B=IP(RAN(3))
1.04 TO STEP 1.09 IF A=8 OR A=10
1.042 DO PART 11 IF A=6 OR B<>1
1.0435 SET C=IP(RAN(2))
1.05 TO STEP 1.07 IF A=9
1.055 DO PART 4 IF C=1
1.06 DO PART 55 IF IP(RAN(2))=1 OR C=1 OR A=7
1.07 DO PART 22 IF A=6 OR B<>1
1.075 DO PART 4 IF A=9 AND B=1
1.08 DO PART 2 IF A=6 OR A=8
1.09 LINE
1.10 TO STEP 1.01
```

```
2.01 DO PART 11
2.02 TO STEP 2.05 IF IP(RAN(3))<>1
2.03 SET C=IP(RAN(3))
2.035 DO PART 4 IF C=1
2.04 DO PART 55 IF C=1 OR IP(RAN(2))=1
2.05 DO PART 55 IF IP(RAN(3))=1
2.06 DO PART 22

4.0 DO STEP IP(RAN(3)+1)/10+4
4.05 DONE
4.1 SAY DEFTLY
4.2 SAY SHREWDLY
4.3 SAY RHYTHMICALLY

6.0 DO STEP IP(RAN(3)+1)/10+6
6.05 DONE
6.1 SAY CONSIDERED
6.2 SAY GAVE
6.3 SAY BELIEVED

7.0 DO STEP IP(RAN(3)+1)/10+7
7.05 DONE
7.1 SAY REMAINED
7.2 SAY BECAME
7.3 SAY WAS

8.0 DO STEP IP(RAN(3)+1)/10+8
8.05 DO PART 55
8.06 DONE
8.1 SAY APPEARED
8.2 SAY SMELLED
8.3 SAY GREW

9.0 DO STEP IP(RAN(5)+1)/10+9
9.05 DONE
9.1 SAY CHARMED
9.2 SAY ANNOYED
9.3 SAY EXPOSED
9.4 SAY FELT
9.5 SAY SQUASHED

10.0 DO STEP IP(RAN(4)+1)/10+10
10.05 DO PART 44
10.06 DONE
10.1 SAY BLUSHED
10.2 SAY WORKED
10.3 SAY CROAKED
10.4 SAY RAN
```

```
11.0 DO STEP IP(RAN(3)+1)/10+11
11.05 DONE
11.1 SAY A
11.2 SAY THE
11.3 SAY THAT

22.0 DO STEP IP(RAN(5)+1)/10+22
22.05 DONE
22.1 SAY FAIRY
22.2 SAY FROG
22.3 SAY FLIRT
22.4 SAY WENCH
22.5 SAY CUR

44.1 TO STEP 44.4 IF IP(RAN(2))=1
44.2 DO PART 4
44.3 DONE
44.4 DO PART 111 IF IP(RAN(4))<>1
44.5 DO PART 2

55.0 DO STEP IP(RAN(5)+1)/10+55
55.05 DONE
55.1 SAY YELLOW
55.2 SAY BRIGHT
55.3 SAY PUTRID
55.4 SAY SLIPPERY
55.5 SAY STICKY

111.0 DO STEP IP(RAN(4)+1)/10+111
111.05 DONE
111.1 SAY TO
111.2 SAY IN
111.3 SAY AT
111.4 SAY FROM
```

Operation of Program (First Transcript):

```
THE
FAIRY
CONSIDERED
THE
FROG

A
FROG
GREW
SLIPPERY
```

THAT
FROG
BECAME
A
FAIRY

THE
SLIPPERY
WENCH
WAS
A
BRIGHT
FLIRT

A
FAIRY
APPEARED

THE
YELLOW
CUR
CROAKED

Operation of Program (Second Transcript):

A
COMPUTER
RAN

A
QUICK AND DIRTY
DOG
CHARMED
THE
SECRETARY

A
COMPUTER
BELIEVED
A
SECRETARY

THAT
PROGRAMMER
GREW
GLORIOUS

A
COMPUTER
GREW
OBSCURE

THAT
COMPUTER
DESIGNED
THAT
PROGRAMMER

THE
SLOW
COMPUTER
FLUSHED
INTENSELY

THAT
DOG
BELIEVED
THE
COMPUTER

THE
COMPUTER
FLUSHED

THAT
COMPUTER
ACTED
GLORIOUS

WORDS - Generation of Words

This program generates four-letter words in accordance with prescribed arrangements of consonants and vowels. Selection of the individual letters is by random, as determined by a random number generator program in the computer.

Listing of Program:

```
1.005 TYPE "THIS PROGRAM GENERATES FOUR LETTER WORDS THAT FIT"
1.006 TYPE "ONE OF THE LETTER PATTERNS:"
1.007 TYPE "   . CVVC   CVCE   CCVC   CVCC   "
1.008 TYPE "C = CONSONANT, V = VOWEL, AND E = THE LETTER E"
1.009 TYPE #,#,#
1.01 A=IP(RAN(4)+2)
1.02 DO PART A
1.03 TYPE #,#
1.04 TO STEP 1.01
1.085 TYPE IP(A),IP(B),IP(C),IP(D)
1.17 TO STEP 1.01
1.29 PRINT "M"
1.30 PRINT "B"
1.31 PRINT "C"
1.32 PRINT "D"
1.33 PRINT "F"
1.34 PRINT "G"
1.35 PRINT "H"
1.36 PRINT "J"
1.37 PRINT "K"
1.38 PRINT "L"
1.39 PRINT "M"
1.40 PRINT "N"
1.41 PRINT "P"
1.42 PRINT "Q"
1.43 PRINT "R"
1.44 PRINT "S"
1.45 PRINT "T"
1.46 PRINT "V"
1.47 PRINT "W"
1.48 PRINT "X"
1.49 PRINT "Y"
1.50 PRINT "Z"
1.51 PRINT "A"
1.52 PRINT "E"
1.53 PRINT "I"
1.54 PRINT "O"
1.55 PRINT "U"
1.56 PRINT "AI"
1.57 PRINT "EA"
1.58 PRINT "EE"
1.59 PRINT "OI"
1.60 PRINT "OO"
1.61 PRINT "OU"
1.62 TYPE #,#
```

```
2.1 DO STEP 1+IP(X)/100 FOR X=RAN(21)+30,RAN(5)+56,RAN(21)+30

3.1 DO STEP 1+IP(X)/100 FOR X=RAN(21)+30,RAN(5)+51,RAN(21)+30
3.2 DO STEP 1.52

4.1 DO STEP 1+IP(X)/100 FOR X=RAN(21)+30,RAN(5)+51,RAN(21)+30
4.2 DO STEP 1+IP(X)/100 FOR X=RAN(21)+30

5.1 DO STEP 1+IP(X)/100 FOR X=RAN(21)+30,RAN(21)+30,RAN(5)+51
5.2 DO STEP 1+IP(X)/100 FOR X=RAN(21)+30
```

Output of Program:

```
THIS PROGRAM GENERATES FOUR LETTER WORDS THAT FIT
ONE OF THE LETTER PATTERNS:
          CVVC    CVCE    CCVC    CVCC
C = CONSONANT, V = VOWEL, AND E = THE LETTER E

QEAS
KIDE
CBIL
CEPE
PAIR
ZOOG
ZEEL
XOSE
QAGE
CICE
CEAT
PEQF
VEPX
YIZE
LEAY
YUQE
KEVE
GUGJ
NOIQ
QKUF
DYIR
HEAR
ZUJE
XORE
MEGE
QOUJ
QAIF
```

DATE - Program to Give Day of Week of any Date

Given a date (day, month, and year, according to the Gregorian calendar) this program determines the day of the week. Can you figure out why the months and days are assigned their particular number values? Notice the Leap Year correction in steps 1.18 and 1.19; also the correction of step 1.16 which takes into account Year "0".

Listing of Program:

```
1.01 SAY ALL DATES REFER TO THE GREGORIAN CALENDAR
1.011 LINE
1.012 LINE
1.02 SET MAR=2
1.03 SET APR=5
1.04 SET MAY=0
1.05 SET JUN=3
1.06 SET JUL=5
1.07 SET AUG=1
1.08 SET SEP=4
1.09 SET OCT=6
1.10 SET NOV=2
1.11 SET DEC=4
1.12 SET JAN=6
1.13 SAY WHAT YEAR (NEGATIVE IF BC)?
1.14 DEMAND YR
1.15 SET S=0
1.16 SET S=1 IF YR<0
1.17 SET YR='YR'
1.18 TO STEP 1.21 IF FP(YR/4)>0
1.19 TO STEP 1.21 IF FP(YR/400)>0 AND FP(YR/100)=0
1.20 SET JAN=5
1.21 SET FEB=JAN-4
1.22 SAY WHAT MONTH (USE THREE LETTER ABREVIATIONS)?
1.23 DEMAND MO
1.24 SAY WHAT DAY OF THE MONTH?
1.25 DEMAND DAY
```

```
1.26 SET X=MO+DAY-IP((MO+DAY)/7)*7
1.27 SET A=IP(YR/100)-IP(YR/400)
1.28 SET A=A-S IF JAN=6
1.29 SET M=YR+IP(YR/4)-A
1.30 SET M=M-IP(M/7)*7
1.31 SET M=7-M IF S=1
1.32 SET Q=IP(M+X-IP((M+X)/7)*7+2.001)
1.321 LINE
1.322 SAY THAT DAY IS...
1.33 DO PART Q
1.332 LINE
1.34 TO STEP 1.12

2 SAY SUNDAY

3 SAY MONDAY

4 SAY TUESDAY

5 SAY WEDNESDAY

6 SAY THURSDAY

7 SAY FRIDAY

8 SAY SATURDAY
```

Operation of Program:

```
ALL DATES REFER TO THE GREGORIAN CALENDAR

WHAT YEAR (NEGATIVE IF BC)?
YR=1966
WHAT MONTH (USE THREE LETTER ABREVIATIONS)?
MO=JUL
WHAT DAY OF THE MONTH?
DAY=4

THAT DAY IS...
MONDAY

WHAT YEAR (NEGATIVE IF BC)?
YR=1965
WHAT MONTH (USE THREE LETTER ABREVIATIONS)?
MO=JUL
WHAT DAY OF THE MONTH?
DAY=4

THAT DAY IS...
SUNDAY

WHAT YEAR (NEGATIVE IF BC)?
YR=1964
WHAT MONTH (USE THREE LETTER ABREVIATIONS)?
MO=JUL
WHAT DAY OF THE MONTH?
DAY=4

THAT DAY IS...
SATURDAY

WHAT YEAR (NEGATIVE IF BC)?
YR=1963
WHAT MONTH (USE THREE LETTER ABREVIATIONS)?
MO=JUL
WHAT DAY OF THE MONTH?
DAY=4

THAT DAY IS...
THURSDAY
```

ORDER - Rank Orders a List of Numbers

This program sorts a given set of integers into a monotone sequence.

Listing of Program:

```
1.01 UP=0, DOWN=1, A=1, S=0, C=0
1.02 READ "HOW MANY NUMBERS? ",N," ORDER UP OR DOWN?  ",ORDER,#
1.03 DEMAND P[I] FOR I=1:1:N
1.04 X=P[I] IF X<P[I] FOR I=1:1:N FOR X=P[1]
1.05 I=1, S=0, C=1 IF S=N @ C=0
1.06 S=S+1 IF P[I]<=P[J] FOR J=1:1:N
1.07 Q[A]=P[I], P[I]=X, A=A+1 IF S=N
1.08 I=I+1, S=0 IF A<N+1 & S<N
1.09 TO STEP 1.05 IF A<N+1
1.095 TYPE #,"YOUR NUMBERS IN DESIRED ORDER ARE:"
1.10 TYPE Q[A] FOR A=N:-1:1 IF ORDER =1
1.11 TYPE Q[A] FOR A=1:1:N IF ORDER =0
```

Operation of Program:

```
HOW MANY NUMBERS? 3  ORDER UP OR DOWN?  UP
        P[1]=4
        P[2]=1
        P[3]=7

YOUR NUMBERS IN DESIRED ORDER ARE:
        Q[1]=    1
        Q[2]=    4
        Q[3]=    7
←DO PART 1
HOW MANY NUMBERS? 5  ORDER UP OR DOWN?  DOWN
        P[1]=6
        P[2]=1
        P[3]=9
        P[4]=23
        P[5]=6

YOUR NUMBERS IN DESIRED ORDER ARE:
        Q[5]=   23
        Q[4]=    9
        Q[3]=    6
        Q[2]=    6
        Q[1]=    1
```

MUSIC - Generates a Random Melody

This simple program for generating a random melody was written by a sixth grade student. The key of B flat was presumed; the numerical code for the notes is as follows.

Ø = rest	5 = F
1 = B flat	6 = G
2 = C	7 = A
3 = D	8 = B flat
4 = E flat	

Listing of Program:

```
1.1 SAY COMPUTER MUSIC BY PDP1
1.2 LET A=IP(RAN(8))
1.3 TYPE A
1.4 LET B=IP(RAN(8))
1.5 TYPE B
1.6 LET C=IP(RAN(8))
1.7 TYPE C
1.8 TO STEP 1.2
```

Output of Program:

```
←DO PART 1 FOR N=1(1)2Ø
COMPUTER MUSIC BY PDP1
          A =     7
          B =     Ø
          C =     1
          A =     Ø
          B =     5
          C =     1
          A =     Ø
          B =     3
          C =     1
          A =     2
          B =     7
          C =     2
          A =     3
          B =     7
          C =     6
          A =     5
          B =     7
          C =     6
          A =     2
          B =     4
          C =     4
          A =     2
```

B = 7
C = 6
A = 4
B = Ø
C = 3
A = 7
B = 6
C = 3
A = 1
B = 1
C = 2
A = 1
B = 4
C = 2
A = 7
B = 6
C = 6
A = 7
B = 2
C = 7
A = 2
B = 3
C = 5
A = 7
B = Ø
C = 1
A = 4
B = 4
C = 6
A = 4
B = 4
C = Ø
A = 5
B = 7
C = 6
A = Ø
B = 3
C = 3
A = 1
B = Ø
C = 1
A = 6
B = 7
C = 4
A = Ø
B = 3
C = 5
A = 5
B = 1
C = 4
A = 4
B = 6
C = 1
A = 7

CANMIS - Cannibal-Missionary Game

This program plays cannibals and missionaries, or rather keeps score for the user as he plays. The game is a test of the player's logic and ability to think ahead.

The appeal that has made this puzzle appear in so many forms and over so long a time span, lies in two things: first, the rules are so simple that anyone can understand them; and secondly, the game can be represented in lively concrete images.

N cannibals and N missionaries are standing on the jungle side of the Limpopo River. Food is running low, and the missionaries are very eager to herd their flock over to the village side, where lots of other things to eat abound. There is, unfortunately, only one small boat; so small in fact, that it will carry no more than two persons. Unfortunately also, the cannibals state of religion is such that, should they ever outnumber the missionaries on one side of the river, they would enjoy a feast at the expense of their guardians. Can you, the head thinker for the missionaries, get everyone safely to the village side of the Limpopo? You are allowed to choose the number of cannibals and missionaries under your care, so long as they are equal.

This program is rather entertaining, and makes a good demonstration piece. It may help stimulate interest in persons who need a boost getting involved with the computer. Its logical value in use goes as far as the game.

The program which plays the game does have some interesting features for the student. The programmer does not have to solve the puzzle in order to write the program. The algorithm which the program performs does a different type of logic. It recognizes all the possible right and wrong attributes an individual move can have. To do so, it poses complex problems in keeping track of the moves, the personnel and the boat. One of these involves a loop; one a value replacement and one a binary operation which in turn controls printout. In addition, one wishes to perform the various tests in such an order as to avoid duplication. This is a logic problem in itself. In solving it, the student sees the difference between an elegant, brief algorithm, and a long messy one. Many mathematicians regard such conciseness as an important aspect of their art.

Listing of Program:

```
61.ØØ5  TYPE #,#,#
61.Ø1 TYPE "THERE ARE K CANNIBALS AND K MISSIONARIES"
61.Ø2 TYPE "ON THE JUNGLE SIDE OF THE RIVER, WISHING"
61.Ø3 TYPE "TO CROSS TO THE VILLAGE.  THE BOAT HOLDS ONLY"
61.Ø4 TYPE "2 PASSENGERS.  IF THERE ARE EVER MORE CANNIBALS"
61.Ø5 TYPE "THAN MISSIONARIES ON ONE SIDE OF THE"
61.Ø6 TYPE "RIVER, THE MISSIONARIES BECOME MISSIONARY STEW."
61.Ø65 TYPE "CAN YOU GET THEM SAFELY ACROSS?",#
61.Ø7 READ, "HOW MANY OF EACH ARE THERE?",K,#,#
61.Ø8 C=4,M=1,N=Ø,MOV=Ø,JC=K,JM=K,CH=1,PR=65
61.Ø9 TYPE "CONGRATULATIONS, YOU DID IT!!!" IF 'JC'+'JM'=Ø
61.1Ø TYPE MOV IN FORM 13 IF 'JC' + 'JM' = Ø
61.11 DONE IF 'JC'+'JM'=Ø
61.12 DO PART 62
61.13 TYPE "THE CANNIBALS HAVE DINNER." IF JC>JM AND JM><Ø
61.135 TYPE "THE CANNIBALS HAVE DINNER." IF K-JC>K-JM AND K-JM><Ø
61.14 DONE IF JC>JM AND JM><Ø OR K-JC>K-JM AND K-JM><Ø
61.15 MOV=MOV+1
61.16 TYPE "TYPE M,C, OR N TO INDICATE THE BOATS PASSENGERS"
61.17 DEMAND PON,PTW
61.18 JC=JC-CH IF PON=4
61.19 JC=JC-CH IF PTW=4
61.2Ø JM=JM-CH IF PON=1
61.21 JM=JM-CH IF PTW=1
61.215 TYPE "SO WHO IS GOING TO ROW?" IF PTW=Ø AND PON=Ø
61.22 DONE IF PTW=Ø AND PON=Ø
61.23 CH=CH*-1
61.24 TYPE "THAT'S IMPOSSIBLE" IF JC<Ø OR JM<Ø OR JC>K OR JM>K
61.25 DONE IF JC<Ø OR JM<Ø OR JC>K OR JM>K
61.26 TO STEP 61.Ø9

62.Ø1 TYPE #,#,"               JUNGLE SIDE",#
62.Ø2 DO PART 64 IF CH=1
62.Ø25 DO PART 65 IF CH=-1
62.Ø3 TYPE #,"---------------RIVER----------------",#
62.Ø4 DO PART 66 IF CH=1
62.Ø45 DO PART 67 IF CH=-1
62.Ø5 TYPE #,"               VILLAGE SIDE",#

64.Ø1 TYPE JC,JM,"BOAT" IN FORM 14

65.Ø1 TYPE JC,JM,"      " IN FORM 14

66.1 TYPE K-JC,K-JM,"     " IN FORM 14

67.1 TYPE K-JC,K-JM,"BOAT" IN FORM 14

FORM  13
YOU SOLVED THE PROBLEM IN ### MOVES
FORM  14
## CANNIBALS    ## MISSIONARIES     ######
```

Output of Program:

```
THERE ARE K CANNIBALS AND K MISSIONARIES
ON THE JUNGLE SIDE OF THE RIVER, WISHING
TO CROSS TO THE VILLAGE.  THE BOAT HOLDS ONLY
2 PASSENGERS.  IF THERE ARE EVER MORE CANNIBALS
THAN MISSIONARIES ON ONE SIDE OF THE
RIVER, THE MISSIONARIES BECOME MISSIONARY STEW.
CAN YOU GET THEM SAFELY ACROSS?
         K=2


               JUNGLE SIDE

2 CANNIBALS     2 MISSIONARIES     BOAT

---------------RIVER---------------

Ø CANNIBALS     Ø MISSIONARIES

               VILLAGE SIDE

TYPE M,C, OR N TO INDICATE THE BOATS PASSENGERS
       PON=N
       PTW=C


               JUNGLE SIDE

1 CANNIBALS     2 MISSIONARIES

---------------RIVER---------------

1 CANNIBALS     Ø MISSIONARIES     BOAT

               VILLAGE SIDE

TYPE M,C, OR N TO INDICATE THE BOATS PASSENGERS
       PON=N
       PTW=N
SO WHO IS GOING TO ROW?
```

```
K=2

                JUNGLE SIDE

2 CANNIBALS      2 MISSIONARIES      BOAT

---------------RIVER---------------

Ø CANNIBALS      Ø MISSIONARIES

                VILLAGE SIDE

TYPE M,C, OR N TO INDICATE THE BOATS PASSENGERS
       PON=M
       PTW=C

                JUNGLE SIDE

1 CANNIBALS      1 MISSIONARIES

---------------RIVER---------------

1 CANNIBALS      1 MISSIONARIES      BOAT

                VILLAGE SIDE

TYPE M,C, OR N TO INDICATE THE BOATS PASSENGERS
       PON=M
       PTW=N

                JUNGLE SIDE

1 CANNIBALS      2 MISSIONARIES      BOAT

---------------RIVER---------------

1 CANNIBALS      Ø MISSIONARIES

                VILLAGE SIDE

TYPE M,C, OR N TO INDICATE THE BOATS PASSENGERS
       PON=M
       PTW=M
```

```
                    JUNGLE SIDE

1 CANNIBALS         Ø MISSIONARIES

---------------RIVER---------------

1 CANNIBALS         2 MISSIONARIES      BOAT

                    VILLAGE SIDE

TYPE M,C, OR N TO INDICATE THE BOATS PASSENGERS
       PON=M
       PTW=N


                    JUNGLE SIDE

1 CANNIBALS         1 MISSIONARIES      BOAT

---------------RIVER---------------

1 CANNIBALS         1 MISSIONARIES

                    VILLAGE SIDE

TYPE M,C, OR N TO INDICATE THE BOATS PASSENGERS
       PON=M
       PTW=C
CONGRATULATIONS, YOU DID IT!!!
YOU SOLVED THE PROBLEM IN   5 MOVES
←
```

NAVIG - A Problem in Navigation

This program is one of many navigation and targeting games written by high-school students. In order to write it, a student must have a thorough knowledge of the mathematics of ballistics, trajectories, and relative velocities; it represents, therefore, a high degree of mathematical sophistication on his part.

Listing of Program:

```
1.1 TYPE "YOU NAVIGATE A BOMBER WHICH SIGHTS AN ENEMY BOAT TWICE"
1.11 TYPE "AT A ONE MIN. INTERVAL.  THE PLANE'S AND BOAT'S POSITIONS"
1.12 TYPE "ARE CONSIDERED AS POINTS ON AN X,Y PLANE.  YOU'RE"
1.13 TYPE "OVER THE ORIGIN DURING THE SECOND SIGHTING."
1.2 TYPE "YOUR HEADING IS CONSTANT.  THE BOMB ACCELERATES DOWNWARD"
1.3 TYPE "AT ONE UNIT/MIN↑2.  FIND YOUR VELOCITY AND THE POINT"
1.4 TYPE "TO RELEASE THE BOMB IN ORDER TO SINK THE SHIP"
1.45 PRINT #
1.5 TO PART 2

2.2 DSA=10+RAN(21)
2.3 A=IP(RAN(DSA+.1))
2.33 SET A=-A IF (RAN(2))<1
2.36 B=IP(SQRT(DSA↑2-A↑2+1))
2.4 SET B=-B IF RAN(2)<1
2.45 PRINT A,"  ",B," ARE THE COORDINATES OF THE 1ST SIGHTING"
2.46 PRINT #
2.5 DSB=12+RAN(21)
2.53 X=IP(RAN(DSB+.1))
2.56 SET X=-X IF (RAN(2))<1
2.57 W=X-A
2.6 Y=IP(SQRT(DSB↑2-X↑2+.1))
2.63 SET Y=-Y IF (RAN(2))<1
2.65 TO STEP 2.5 IF W=0 OR X=0 OR A=0
2.66 TO STEP 2.5 IF B/A=Y/X OR X=2*A OR Y=B
2.68 MST=(Y-B)/W
2.69 XM=X+.5*W
2.7 YM=Y+.5*(Y-B)
2.71 TO PART 9 IF (YM/XM)<(Y/X)
2.73 MY=RAN((YM/XM)-(Y/X))+(Y/X)
2.74 I=1
2.75 MA=(.1*(IP(10*MY)))+I*.1
2.76 TO STEP 2.5 IF MA=0
2.78 TO STEP 2.5 IF (MA/MST)>.9 AND (MA/MST)<1.1
2.82 PRINT X,"  ",Y," ARE COORDINATES OF 2ND SIGHTING 1 MIN. LATER"
2.83 PRINT #
2.92 PRINT MA," IS SLOPE OF BOMBER'S HEADING FROM (0,0)←TARGET"
2.93 PRINT #,#
```

```
2.97 TYPE "FIND THE SPEED (VP) TO MAINTAIN FROM"
2.975 TYPE "(0,0)←TARGET AND (DX,DY)AT WHICH YOU RELEASE BOMB"
2.98 PRINT #,#,#
2.99 TO PART 3

3.1 QQ=MST/MA
3.13 YI=[(X*B-A*Y)/(W*(1-QQ))]
3.16 XI=YI/MA
3.18 E=(XI↑2+YI↑2)*(W↑2+((Y-B)↑2))
3.2 PV=SQRT(E/((XI-X)↑2+(YI-Y)↑2))
3.22 L=(RAN(1/4)+(1/2))
3.24 OD=[SQRT(XI↑2+YI↑2)*L]
3.26 F=((1-OD)/PV)
3.27 H=F↑2/2
3.3 PRINT H,"   IS YOUR CONSTANT ALTITUDE IN UNITS"
3.42 XD=OD*XI/(SQRT(XI↑2+YI↑2))
3.44 YD=YI*XD/XI
3.45 PRINT #
3.47 DEMAND DX
3.475 DEMAND DY
3.48 DEMAND VP
3.5 PRINT #
3.6 TO PART 4 IF DX<XD-.5 OR DX>XD+.5
3.65 TO PART 4 IF DY<YD-.5 OR DY>YD+.5
3.7 TO PART 5 IF VP/PV>.96 AND VP/PV<1.04
3.8 TO PART 4

4.1 TYPE "U MIST"
4.12 PRINT #
4.15 TYPE "(XD,YD) IS CORRECT POINT--PV IS CORRECT SPEED"
4.16 PRINT #
4.2 TYPE PV,XD,YD
4.3 DO PART (10+(IP(RAN(7))))

5.1 TYPE "BAVOOM--NOT BAD FOR AN AMATEUR"

9.1 MY=RAN((Y/X)-(YM/XM))+(YM/XM)
9.2 I=-1
9.3 TO STEP 2.75
```

Output of Program:

```
←DO PART 1
YOU NAVIGATE A BOMBER WHICH SIGHTS AN ENEMY BOAT TWICE
AT A ONE MIN. INTERVAL.  THE PLANE'S AND BOAT'S POSITIONS
ARE CONSIDERED AS POINTS ON AN X,Y PLANE.  YOU'RE
OVER THE ORIGIN DURING THE SECOND SIGHTING.
YOUR HEADING IS CONSTANT.  THE BOMB ACCELERATES DOWNWARD
AT ONE UNIT/MIN↑2.  FIND YOUR VELOCITY AND THE POINT
TO RELEASE THE BOMB IN ORDER TO SINK THE SHIP

-9  -19  ARE THE COORDINATES OF THE 1ST SIGHTING
7  -17  ARE COORDINATES OF 2ND SIGHTING 1 MIN. LATER
-.9  IS SLOPE OF BOMBER'S HEADING FROM (0,0)←TARGET

FIND THE SPEED (VP) TO MAINTAIN FROM
(0,0)←TARGET AND (DX,DY)AT WHICH YOU RELEASE BOMB


.0611466   IS YOUR CONSTANT ALTITUDE IN UNITS
           DX=-10
           DY=9
           VP=1.6

U MIST

(XD,YD) IS CORRECT POINT--PV IS CORRECT SPEED

           PV=   35.96015
           XD=   10.09054
           YD=   -9.081482
NEXT TIME BRING A PARACHUTE
←
```

Students have written programs to play several games, such as tic-tac-toe, 21, chess, and the like, as well as games of their own invention. Game-playing problems are a source of interest to them and serve as an excellent vehicle for the study of mathematical strategies and the algorithmic approach to solving problems. The following program, illustrative of such efforts, plays chemin de fer, the game at which James Bond was an expert.

Listing of Program:

```
1.001 SAY THE OBJECT OF CHEMIN DE FER IS TO GET AS CLOSE TO
1.002 SAY 9 POINTS AS POSSIBLE. FACE CARDS AND 10'S COUNT
1.003 SAY 10 OR 0. TWO CARDS ARE DEALT, A THIRD IF WANTED,
1.004 SAY IF FIRST 2 COUNT 8 OR 9 THEY ARE A NATURAL; 9 BEATS 8.
1.005 SAY IF THE SUM IS GREATER THAN 10 ONLY UNITS COUNT,15=5
1.006 SAY A WILL REPRESENT YOUR INDIVIDUAL CARDS; I THEIR SUM
1.012 SET AA=IP(RAN(12))
1.015 SET AS=IP(RAN(12))
1.016 TYPE AA,AS
1.02 SET AA=0 IF AA>10 OR AA=1
1.03 SET AS=0 IF AS>10 OR AS=1
1.04 TYPE AA,AS
1.06 SET I=AA+AS
1.07 SET I=I-10 IF I>10
1.08 DO PART 4 IF I=8 OR I=9
1.085 SAY YOUR POINTS =I
1.09 TYPE I
1.1 SAY TYPE 0 IF YOU DON'T WANT ANOTHER CARD.
1.11 DEMAND X
1.12 DO PART 2 IF X=0
1.13 SET AD=IP(RAN(12))
1.14 SET AD=0 IF AD=1 OR AD>10
1.15 SET I=AA+AS+AD
1.16 SET I=I-10 IF I>10
1.17 TO STEP 1.16 IF I>10
1.18 TYPE I
1.19 TO STEP 2.0

2.000 SET JJ=IP(RAN(12))
2.001 SET JK=IP(RAN(12))
```

```
2.002 TYPE JJ, JK
2.02 SET JJ=0 IF JJ>10 OR JJ=1
2.03 SET JK=0 IF JK=1 OR JK>10
2.04 SAY L=SUM OF DEALERS CARDS.
2.05 SET L=JK+JJ
2.06 SET L=L-10 IF L>10
2.07 TO STEP 2.06 IF L>10
2.075 TYPE L
2.08 DO PART 3 IF L<I
2.09 TO STEP 4.07 IF I=9
2.1 TO STEP 4.5 IF I=8 AND L=10
2.12 SAY YOU LOSE!
2.13 TO STEP 1.012

3.0 SET JL=IP(RAN(12))
3.01 TO STEP 3.02
3.02 SET JL=0 IF JL>10 OR JL=1
3.025 TYPE JL
3.03 SET L=JJ+JK+JL
3.04 SET L=L-10 IF L>10
3.05 TO STEP 3.04 IF L>10
3.06 TO STEP 4.07 IF L<I
3.07 SAY YOU WIN!
3.08 TO STEP 1.012

4.0 SET JJ=IP(RAN(12))
4.01 SET JK=IP(RAN(12))
4.02 SET JJ=0 IF JJ=1 OR JJ>10
4.03 SET JK=0 IF JK=1 OR JK>10
4.04 SET L=JK+JJ
4.05 SET L=L-10 IF L>10
4.06 TO STEP 4.05 IF L>10
4.07 TO STEP 4.5 IF L=9 AND I=9
4.08 TO STEP 4.1 IF L=9 AND I=8
4.09 SAY YOU WIN!
4.095 TO STEP 1.012
4.1 SAY YOU LOSE
4.2 TO STEP 1.012
4.5 SAY IT'S A DRAW.
4.6 TO STEP 1.012

*
```

Operation of Program:

```
THE OBJECT OF CHEMIN DE FER IS TO GET AS CLOSE TO
9 POINTS AS POSSIBLE. FACE CARDS AND 10'S COUNT
10 OR 0. TWO CARDS ARE DEALT, A THIRD IF WANTED,
IF FIRST 2 COUNT 8 OR 9 THEY ARE A NATURAL; 9 BEATS 8.
IF THE SUM IS GREATER THAN 10 ONLY UNITS COUNT,15=5
A WILL REPRESENT YOUR INDIVIDUAL CARDS; I THEIR SUM
          AA =    11
          AS =     1

          AA =     0
          AS =     0

YOUR POINTS =I
           I =     0

TYPE 0 IF YOU DON'T WANT ANOTHER CARD.
X=65
           I =     2

          JJ =     0
          JK =     8

L=SUM OF DEALERS CARDS.
           L =     8

YOU LOSE!
          AA =     2
          AS =     0

          AA =     2
          AS =     0

YOUR POINTS =I
           I =     2

TYPE 0 IF YOU DON'T WANT ANOTHER CARD.
X=1
           I =     7

          JJ =     2
          JK =     3

L=SUM OF DEALERS CARDS.

           L =     5

          JL =    10

YOU WIN!
```

PLANES - Exercise in Probability

This program makes a flashy and entertaining game. It can be used to catch the eye at demonstrations, to interest uninformed or indifferent persons, or to show the speed and range of the computer. More seriously, the game can be used to introduce a student to game theory and probability. A very interesting task can be posed by showing a pupil the game in action, and asking him to write the program which is playing it.

The player and the machine each fly a squadron of P planes, carrying B bullets apiece, P and B to be chosen by the player. The two squadrons make runs at one another, and each squadron can fire once during each run. The firing positions are near, middle and far, and the player is allowed to choose the probabilities of a hit from each distance. All the planes in a squadron must fire at once.

After each salvo, the computer will calculate the number of remaining planes and bullets per side, and type them. The game is over when either side has lost all his planes or when both sides have run out of bullets. Since the computer plays an honest game by the laws of probability, a skillful player can beat it one game in two. A guesser, needless to say, is not likely to do so well.

Listing of Program:

```
1.00 TYPE "YOU AND I ARE EACH FLYING A SQUADRON OF P PLANES"
1.01 TYPE "CARRYING B BULLETS APIECE, P AND B TO BE CHOSEN BY YOU."
1.02 TYPE "THE TWO SQUADRONS MAKE RUNS AT ONE ANOTHER, AND"
1.03 TYPE "EACH SQUADRON CAN FIRE ONCE DURING EACH RUN.  THE FIRING"
1.04 TYPE "POSITIONS ARE CLOSE, MIDDLE, AND FAR.  YOU ARE "
1.05 TYPE "ALLOWED TO CHOOSE THE PROBABILITIES OF A HIT FROM EACH"
1.06 TYPE "DISTANCE.  ALL THE PLANES IN A SQUADRON MUST FIRE AT ONCE."
1.07 DO PART 51
```

```
51.01 TYPE #,#,#,"HOW MANY PLANES AND BULLETS SHALL WE EACH HAVE?"
51.02 DEMAND PLN,BUL
51.03 TYPE"WHAT ARE THE PROBABILITIES OF A HIT FROM EACH DISTANCE?"
51.04 DEMAND FAR,MID,CLO
51.05 P[1]=FAR,P[2]=MID,P[3]=CLO,M=PLN,Y=M,MB=BUL,TYP=0
51.06 YB=MB,YES=1,NO=0,ME=M
51.065 F=0,D=1,ANS=0,CK=0
51.07 L[1]=M*P[1]*(3+P[2]+P[3])-Y*(P[1]+P[2]+P[3])
51.08 L[2]=M*P[2]*(3+P[3])-Y*(P[1]+P[2]+P[1]+P[2]+P[3])
51.09 L[3]=M*3*P[3]-Y*(P[1]+P[3]+P[2]+P[3]+P[1]+P[2]+P[3])
51.10 C=K IF L[K]=MAX(L[1],L[2],L[3]) FOR K=1:1:3.
51.11 TYPE #,#
51.12 TYPE "YOU",Y,YB IN FORM 10 IF TYP=1
51.13 TYPE "I",M,MB IN FORM 10 IF TYP=1
51.134 TYPE #
51.135 TYP=0
51.14 TYPE #,"IT'S A TIE!" IF Y=M AND YB=0 AND MB=0
51.15 TYPE #,"YOU WON THE BATTLE." IF Y>M AND MB=0 OR Y>M AND M=0
51.16 TYPE #,"I WON THAT ROUND." IF M>Y AND YB=0 OR M>Y AND Y=0
51.17 DONE IF MB=0 AND YB=0 OR M=0 OR Y=0
51.18 TYPE "FAR" IN FORM 11 IF D=1
51.19 TYPE "MIDDLE" IN FORM 11 IF D=2
51.20 TYPE "CLOSE" IN FORM 11 IF D=3
51.21 READ "WILL YOU FIRE NOW?  ",ANS,# IF F=0 AND YB>0
51.225 TO STEP 51.26 IF CK=1 & D=2
51.23 TYPE #,"I FIRED THIS TIME." IF C=D @ CK=1 & D=3
51.24 DO PART 52 IF YB>0 AND ANS=1 AND F=0
51.25 DO PART 55 IF C=D & MB>0 @ CK=1 & D=3
51.26 M=ME,D=D+1
51.27 TO STEP 51.11 IF D=2 OR D=3
51.28 TO STEP 51.065

52.1 DO PART 53 FOR I=1:1:Y
52.2 YB=YB-1,F=1,TYP=1
52.3 ME=0 IF ME<1
52.4 CK=D

53.1 DO PART 54
53.2 ME=ME-1 IF Z=0

54.1 Z=1
54.2 Z=0 IF RAN(10)<P[D]*10

55.1 DO PART 56 FOR I=1:1:M
55.2 MB=MB-1,TYP=1
55.3 Y=0 IF Y<1

56.1 DO PART 54
56.2 Y=Y-1 IF Z=0
```

Output of Program:

```
HOW MANY PLANES AND BULLETS SHALL WE EACH HAVE?
         PLN=10
         BUL=6
WHAT ARE THE PROBABILITIES OF A HIT FROM EACH DISTANCE?
         FAR=.1
         MID=.3
         CLO=.5


THE DISTANCE IS NOW FAR
WILL YOU FIRE NOW?  NO


THE DISTANCE IS NOW MIDDLE
WILL YOU FIRE NOW?  NO

I FIRED THIS TIME.


YOU HAVE    5 PLANE(S), EACH WITH   6 BULLET(S).
I   HAVE   10 PLANE(S), EACH WITH   5 BULLET(S).

THE DISTANCE IS NOW CLOSE
WILL YOU FIRE NOW?  YES


YOU HAVE    5 PLANE(S), EACH WITH   5 BULLET(S).
I   HAVE    8 PLANE(S), EACH WITH   5 BULLET(S).

THE DISTANCE IS NOW FAR
WILL YOU FIRE NOW?  NO


THE DISTANCE IS NOW MIDDLE
WILL YOU FIRE NOW?  NO

I FIRED THIS TIME.
```

```
YOU HAVE   2 PLANE(S), EACH WITH   5 BULLET(S).
I   HAVE   8 PLANE(S), EACH WITH   4 BULLET(S).

THE DISTANCE IS NOW CLOSE
WILL YOU FIRE NOW?  YES


YOU HAVE   2 PLANE(S), EACH WITH   4 BULLET(S).
I   HAVE   6 PLANE(S), EACH WITH   4 BULLET(S).

THE DISTANCE IS NOW FAR
WILL YOU FIRE NOW?  NO



THE DISTANCE IS NOW MIDDLE
WILL YOU FIRE NOW?  YES


YOU HAVE   2 PLANE(S), EACH WITH   3 BULLET(S).
I   HAVE   6 PLANE(S), EACH WITH   4 BULLET(S).

THE DISTANCE IS NOW CLOSE

I FIRED THIS TIME.


YOU HAVE   0 PLANE(S), EACH WITH   3 BULLET(S).
I   HAVE   6 PLANE(S), EACH WITH   3 BULLET(S).


I WON THAT ROUND.
←
```

APPENDIX III

Section B: Program Library Listings

The listings shown in this section have been selected to show the variety of programs written by students and teachers and stored in a central file in the computer system. Programs illustrating a particular mathematical concept, developed at one of the participating schools, were made available to all other schools through this program library. A high degree of competition developed between students, regardless of the geographical separation of their schools as they used each other's programs and proceeded to write new and better ones. It was not unusual to see a program develop through several degrees of sophistication as first one student, and then another, modified and improved on it to make it more effective.

Two sets of library file listings are shown. The number of programs included increases from the early to the later phase of the project, as well as the variety of topics under development. About twenty percent of the contents of the library file changed from one week to another as unsuccessful programs were discarded and new ones were written and filed.

Program Library Listings, Early Phase of Project

LEXINGTON HIGH SCHOOL

QUIZ	TEACHES FUNCTION EVALUATION
SOLVE	EQUATION SOLVER
LOGIC	GENERATES LOGICAL TRUTH TABLES
EQS.	EQUATION SOLVER
BANDIT	SLOT MACHINE
EQS	BEING WORKED ON
REACTI	HEAT OF REACTION
CEM15	BEING WORKED(?) ON
FUNC	GUESSES FUNCTION MADE UP BY STUDENT
INEQUA	SOLVING QUADRATIC INEQUALITY
LETTER	BIENG WORKED ON
MOLES	FINDING MOLES
CHEMIS	BEING WORKED ON
GASJOR	BEING WORKED ON
ENERGY	BALENCE A EQUATION
HOOP	PLAYS BASKETBALL (DO PART 10)
MUSIC	BEING WORKED ON
ROULET	TYPING
R. WOOD	BEING WORKED ON
FOOTBL	PLAYS FOOTBALL GAME

PHILLIPS ACADEMY

PRIME	NO
SOPHOC	RNDMSTS
TRISOL	SOLVES TRIANGLES!
PØKER3	NOT DEBUGGED
LINEQN	NOT DEBUGGED
GRAF	GRAPHS ANY FUNCTION IN STEP 2.2
VERMN1	VERIFIES MENDELIAN MONOHYBRID RATIO
GEOLIM	APPROX. OF CIRCUM. AND AREA OF CIRCLE
STRATY	GAME PLAYING. SOLUTIONS BY MATRICES
REA	OND
VERMN4	SAME FOR TRIHYBRIDS
T-T-T	PLAYS TIC-TAC-TOE
SERIES	COUNTS
VERMN2	VERIFIES MENDELIAN DIHYBRID RATIO
VERMN3	VERIFYING INCOMPLETE DOMINANCE RATIO
WMS	NOT BUGGED
SINDIV	SYNTHETIC DIVISION
ADD	MULTIPLE PRECISION ADDITION
ROOT	FINDS NTH ROOT OF N CORRECT TO 10↑-3
MIKE Q	GRAPHING POLYNOMIALS
PLOT	TABLE FOR PLOTTING FUNCTIONS
SOLVE1	EXPLAINS SOLVE2
NUUM	PLAYS NIM
T?T?T	INCOMPLETE

Program Library Listings, Late Phase of Project

LEXINGTON HIGH SCHOOL

SYNTH	JH TRIG SYNTH
FRORDR	** ORDERS BY FREQ CNT.
ORDER	** ORDERING
WORDS	** WORD GENERATOR
LETTER	** MIND READING
PRFCTR	** PRIME FACTORS
SENTEN	** SENTENCE GENERATOR
FCTRL	** FACTORIAL
FIBON	** FIBONACCI SERIES
QUAD	** SOLVES QUAD. EQS.
LINEAR	** FINDS LINEAR EQ.
ATTEND	** ATTENDANCE REG.
BASES	** BASE 10 TO BASE N
YOUGES	** NUMBER GUESSING
TIC	GB BWO TIC TAC TOE
CHECK	WK CHECKER
INTERS	CC BWO SET MESS
MASS	ANDY BWO BSE N TO 10
MUSIC	JH WRITES+PLOTSGUD
ANDY	ANDY BWO BSE CHNG
CHRIS	CC BWO DECIMALS
FUGG	SB BWO DECIMALS
SIXTH	RS BWO LENSES
PYWD	PYW BWO FRAC
PYWF	PYW BWO FRAC
BOP	MS BWO FUDGE
JEFFD	BWO RPT DEC.
FONC	JR E BWO CONIC
BOBY	BWO DEC RA
LEITZ	HKA E BWO CONIC SECT.
ANDRE	ANDY BWO TRIANGLES
BOBBY	RA BWO
REPTER	** FRACS TO DECIMS.
PYWZ	PYW BWO FRACTIONS
BEED	GB BWO RPT DECS.
NITE	JH BWO PI
ROMANS	ANDY BWO ROMAN NUMERALS
ROMAN	ANDY BWO DEC TO ROM
BIDDY	DB BWO ROULETTE
DECI	HB RS BWO REP DEC
ROGCAL	ROG BWO CAL MACHINE
MC	MC BWO DECIMAL
MONSTR	CC BWO WORKS 6
MM	MC BWO DECIMALS
CONICS	JR E BWO CONIC
INTEGR	** SIMP+TRAP INTEGRALS
TOFRAC	** DECIMAL TO FRACTION
FRCTNS	MJ BAD RPTNG DCMLS
ROGFRA	ROG BAD REPT. DECS
DCIMAL	BWO PMW FRACTIONS
FAIR	EJ BWO TRIANGLES
PHILA	PF BWO 111000...
DEF	MJ BWO SPECIAL PRIMES
PLOTER	** PLOTS ANY F(X)
AGFA	HKA E BAD CONIC SECT.
IMAGES	** PHYSICS OF MIRRORS

APPENDIX IV

Early Ramifications of Project H-212

Section A. Project Visitations

Section B. Dissemination of Information

Section C. Outgrowths of Project H-212

Early Ramifications of Project H-212

Section A. Project Visitations

From the initiation of the in-school experimental phase of the project in September 1965, strong and continuing interest in the program was expressed by large segments of the educational community. Unanticipated during the initial planning for the project, the requests for visitations to the participating schools to observe the classroom activities became so numerous as to require substantial commitments of time by the project staff. During the 1965-1966 school year more than one hundred individual visitors from thirty-two states and ten foreign countries were given the opportunity to familiarize themselves with the project activities through direct observation of the use of the computer as a teaching tool for instruction in mathematics.

In addition to the on-site visitations, a large number of requests have been received from many individuals and educational agencies for informational releases concerning Project H-212. Copies of the newsletter for the project containing a summary of the hypotheses, objectives, and procedures followed in the research program, as well as selected examples of computer programs written by students and teachers at the three grade levels, have been sent out as far as the supply of the documents has lasted. Steps are being taken to reprint certain of the newsletters for further distribution to meet a continuing flow of requests for information.

Readers of this report may obtain copies of the newsletters by writing to: Director, Computer Mathematics Project H-212
Massachusetts Department of Education
Division of Research and Development
Olympia Avenue
Woburn, Massachusetts 01801

Section B. Dissemination of Information

The dissemination of information through individual or group visitations, and through the publication and distribution of newsletters, has proven to be a much more extensive, and, it is felt, more significant part of the activities of the project than was originally anticipated. It reflects a substantial and growing interest on the part of the educational community in the potential application of modern technology to the improvement of the instructional process. As observed by the staff of Project H-212, the interest fell into four general categories: local educational agencies planning for innovative projects under Title III, ESEA, Public Law 89-10; individual school systems planning to make use of computers as instructional aids with local funding support; state education agencies planning for the introduction of innovative educational practices into the local school systems under their jurisdiction; institutions of higher learning interested in exploring the techniques of using computers in instruction. The following brief summary gives specific examples of the variety of these expressions of interest, and indicates the resulting outcomes in the form of local developmental programs in the use of computers in instruction.

Section C. Outgrowths of Project H-212

1. The Connecticut Department of Education, through use of Title V ESEA funds sponsored a pilot project, designed along the lines of Project H-212 and utilizing instructional materials and methods developed by it, to explore the use of the computer for teaching mathematics in high schools in the towns of Bloomfield, Manchester, and Meriden, Connecticut. A major Title III ESEA project, Metropolitan Effort Toward Regional Opportunity (METRO), representing 28 school systems in Hartford County, was the co-sponsor of the pilot project.

The three local school systems provided space for computer terminal facilities in secondary school classrooms. In August 1966, two teachers from each participating school attended a one-week intensive training session conducted by the staff and consultants for Project H-212. The seminars were continued regularly during the ensuing academic year. Observation of the pilot project was conducted by Connecticut Department of Education personnel and mathematics teachers and administrators from other school systems. Teachers and students in the participating schools used the computer terminals in three ways: first, to assist in the discovery approach to teaching mathematics, to help motivate students, and to develop the understanding of mathematical concepts; second, to emphasize the analytical development of problems into sub-problems; third, to de-emphasize the routine use of drill in mathematics teaching.

2. In Hinesburg, Vermont, the Champlain Valley Union School District initiated a Title III ESEA project based on the Project H-212 program. A grant was received in October 1966, and in-school activities were begun shortly thereafter in Champlain Valley and in Rice Memorial High School in nearby Burlington, Vermont.

Pilot classes in each high school made use of a time-shared computer facility in Cambridge, Massachusetts, through remote teletypewriter terminals connected to the computer through telephone lines. Instructional materials and methods developed in Project H-212 were used in the classroom instruction in mathematics. In addition to this basic program, the project was given the flexibility to permit interested non-academic students, junior high school students and teachers, and interested parents and adults to participate in computer-oriented instruction in mathematics, or to view demonstration classes.

3. The Westwood, Massachusetts school system, one of the participants in Project H-212, has applied for and received a planning grant under Title III ESEA for a project which is an extension of the research effort of H-212 to the demonstration phase. The project, known as LOCAL (Laboratory for Computer-Assisted Learning), includes among its member school systems Lexington, another of the H-212 participants, and three other adjoining systems, Natick, Needham, and Wellesley. The purpose of project LOCAL is to show to as wide an audience as possible the feasibility of using the computer in mathematics instruction.

Since the cost of continuous on-line connection to a computer is still relatively high, this factor is somewhat of a deterrent to more widespread use of time-shared computer services by schools. Project LOCAL is investigating the use of teletypewriter terminals in the "off-line" mode, in which a student prepares a punched paper tape as he writes his program on the machine; this is done with no connection to the computer. When the program is written (punched on tape), the student connects the terminal to the computer, reads in his prepared program, and the computer executes it. By doing this, the relatively long time needed for preparation of the program does not tie up the computer with routine mechanical work.. A saving of up to eighty percent in computer operating time is anticipated.

The project LOCAL program provides computer-oriented instruction for students at grades 10, 11, and 12, and intensive in-service courses for teachers in the utilization of computers as instructional aids.

4. The Framingham, Massachusetts school system, in collaboration with three of the Commonwealth's State Colleges, has received a Title III ESEA planning grant to investigate the use of Project H-212 materials and methods in the context of pre-service training of teachers in the use of educational technology for instruction in mathematics and science. Consideration is also being given to the improvement of educational programs for the terminal student in high school, a matter of growing concern on the part of the educational community.*

This project, known as CCC (Communities, Colleges, Collaboration) is a unique effort designed to provide teachers in

---

* As the program develops from the planning phase into operational status, it will involve some twenty-four public and private schools and ten or more institutions of higher learning.

training in their undergraduate years with actual classroom experience in the use of computers, and other aspects of modern technology, as effective tools for the improvement of instruction. It is anticipated that a close degree of collaboration will develop between the public schools and the teacher preparation institutions, and that both agencies will benefit from the extensive laboratory experience that will be provided for the state college students who will soon be entering the teaching profession.

5. Approximately thirty school systems in Massachusetts have directed more than casual expressions of interest to project personnel, inquiring into means of getting computer terminals into their classrooms for experimentation with their use as instructional aids. A substantial proportion of these school agencies are planning to obtain major funding support from their local school budgets to help towards achieving this purpose.

One of the participants in Project H-212, the Winchester, Massachusetts, school system, has continued the use of a computer terminal in the sixth grade of the Vinson-Owen School beyond the end of the grant period. It was at this school that significant work was done in the H-212 program in the use of the computer for instruction in the relatively difficult (to sixth-graders) concept of scientific notation. (See text of report for a more detailed description of this aspect of H-212.)

The administrative and teaching staff of the Vinson-Owen School felt that such significant progress had been made with the use of the computer in sixth grade mathematics classes that a discontinuance of its use would adversely affect their instructional program. Accordingly, when the H-212 program was terminated, the school board voted to extend full local support for the continuation of computer services to the school.

6. Several state and community institutions of higher learning have become interested in the use of Project H-212 techniques of using computers in instruction. As an example, Dartmouth College at Hanover, New Hampshire, initiated a proposal to the National Science Foundation directed toward a broad exploration of such use in schools in several states in New England. A grant was received by Dartmouth to initiate the program in the summer of 1967 with an intensive

in-service program for teachers from the participating schools. Altogether, computer terminals are now in use in eighteen public and private secondary schools in five of the six New England states for instruction in the mathematical sciences. As in Project H-212, time-shared services are provided from a central time-shared computer facility (at Dartmouth) to the participating schools through leased telephone lines. New instructional materials are under development, and will be disseminated to the educational community as they become available.

7. Information has been received from other states that representatives of school districts and private schools who had visited Project H-212 have taken action to initiate similar programs to explore the use of the computer as an instructional tool. Some are planned as Title III ESEA projects, others with funding support from other sources, local, federal, or private foundations. A detailed summary of such activities is not within the scope of this report; however, it is accurate to say that the dissemination of information concerning Project H-212, whether by direct visitation or by printed documentation, has served to stimulate investigation of the use of computers for the improvement of instruction in a significantly large number of instances, both in this country and abroad.

APPENDIX V

Summary of TELCOMP Language

TELCOMP LANGUAGE SUMMARY

TELCOMP LINE

STEP NUMBER	COMMAND	ARGUMENTS	CLAUSES	COMMENT	TERMINATOR
1.15	TYPE	A, B, C	IN FORM 1 FOR K=1:1:N	;OUTPUT RESULTS	←

DEFINITIONS

DIRECT COMMAND: STEP number omitted; TELCOMP executes command immediately.
STORED COMMAND: STEP number present; stored away as a STEP in a program.
STEP: A stored command; STEP numbers range from 1.00000 to 99.99999.
PART: A group of STEPs whose numbers have same integer part. PART numbers range from 1 to 99.
PROGRAM: A group of STEPs.

FILE

A body of information stored within TELCOMP under a name supplied by user. FILE can contain STEPs, PARTs, variables, FORMs. FILE name can be any combination of up to six letters and digits, first of which is a letter. DUMP creates FILEs: LOAD retrieves their contents; TYPE yields summary information about FILEs, DELETE removes them.

GLOSSARY OF COMMANDS

DIRECT OR STORED

TYPE TELCOMP types indicated numbers, quoted text, expressions, blank line (#), variables, STEPs, PARTs, FORMs. ALL can modify the last four.

```
TYPE 1, B=C/2, "TEXT", #,#
TYPE STEP 1.3, PART 2, FORM 1
TYPE ALL STEPS
TYPE ALL PARTS, ALL FORMS, ALL VALUES
TYPE ALL
     (the last two are equivalent)
```

TELCOMP can type summary information about FILEs. If FILE name specified, length in blocks, date of origin, user number, job number are typed. If ALL FILES specified, list of names is typed.

```
TYPE FILE ABC
TYPE ALL FILES
```

PRINT Exactly like TYPE except TELCOMP omits line advance after PRINTing an argument.

SET Defines variables, must be followed by: variable name = expression. "SET" can be omitted.

```
SET A=1, B=C, D=E+F/2
XYZ = A, GHQ=XYZ*D
```

DO TELCOMP "does" indicated PARTs and STEPs, then returns to user if direct, to next command if stored.

```
DO PART 3
DO STEP 1.7, PART 5, STEP 4.6
```

Fractional PART number: TELCOMP begins at indicated or next higher STEP:

```
DO PART 1.6, PART 3.52
```

LINE TELCOMP sends a line feed.

PAGE TELCOMP advances to next page.

SEND TELCOMP transmits indicated codes to teletype. Codes must be integers between 1 and 255, can be designated by numbers, variables, expressions:

```
SEND 1, 255, A, B+C/E
```

PLOT TELCOMP plots indicated variables and expressions. Restriction: $-1 \leq$ value to be plotted ≤ 1.

```
PLOT -1, 0, 1, A, B, C+D/E
```

DELETE TELCOMP removes indicated variables. STEPs, PARTs, FORMs, FILEs. "ALL" can be used as in TYPE.

```
DELETE A, STEP 1.3, PART 2, FORM 1, FILE A1
DELETE ALL STEPS, ALL VALUES
DELETE ALL PARTS, ALL FORMS, ALL VALUES
DELETE ALL
      (the last two are equivalent,
       DELETE ALL doesn't remove FILES)
```

DEMAND TELCOMP types name of indicated variable, pauses for user to type in value (number, defined variable, expression).

```
DEMAND A, XYZ, GHQ
```

READ Like DEMAND except values read from paper tape prepared off-line.

LOGOUT Disconnects a user from TELCOMP.

DIRECT

DUMP TELCOMP stores indicated variables, STEPs, PARTs, FORMs in file specified. If no file specified, TELCOMP punches indicated material on paper tape and types a copy. ALL can be used as in TYPE.

```
DUMP ALL AS ABCDEF
DUMP A, STEP 1.3, PART 2
DUMP ALL STEPS AS B3
DUMP ALL PARTS, ALL FORMS, ALL VALUES
```

LOAD TELCOMP moves contents of specified file to active part of system. If no file specified, TELCOMP reads in a paper tape prepared by DUMP (or off-line) and types contents.

LOGIN Gives user access to TELCOMP. Must be followed by user identification number. ID may be followed by project number, date, or comment.

```
LOGIN X37ABC    JOB NUMBER
LOGIN AB7133    L7.116/2S
```

GO Error or break causes TELCOMP to continue with program after an interruption caused by hitting the BREAK key or a stored STOP command. GO can also be used after interruption introduced by an error comment if the error is first corrected.

REPEAT TELCOMP repeats entire STEP at which it was interrupted.

EDIT TELCOMP types indicated STEP or FORM, then enters EDIT mode.

```
EDIT STEP 1.7
EDIT FORM 3
```

FORM After form number and terminator TELCOMP pauses for user to enter a FORM.

STORED

TO TELCOMP transfers to indicated PART or STEP. TO changes effective sequence of commands, has effect of modifying the DO command.

```
TO PART 5
TO STEP 4.6
TO PART 4.6
```

DONE TELCOMP skips remaining STEPs in current PART.

STOP TELCOMP STOPs. Control transfers to user.

ALGEBRAIC OPERATORS

+ - * / ↑

PRECEDENCE — Operations performed left-to-right except: exponentiation before another operation to its left; multiplication or division before addition or subtraction to its left. Order of operations changeable by parentheses or square brackets.

MATHEMATICAL FUNCTIONS

SQRT(A)	Positive square root	ATN(A,B)	Arctan (A/B) radians	DP(A)	Digit part
LOG(A)	Base 10 logarithm	ATN(A)	Arctan (A) radians	XP(A)	Exponent part
LN(A)	Natural logarithm	MAX(A,B,C)	Maximum of a list	SGN(A)	Sign of A -1,0,+1
EXP(A)	e raised to the power of A	MIN(A,B,C)	Minimum of a list	SIN(A)	Sine (A) A in radians
'A'	Absolute value	IP(A)	Integer part	COS(A)	Cosine (A) A in radians
		FP(A)	Fraction part		

MODIFYING CLAUSES

FOR — Command is executed repeatedly for specified range and/or list of values FOR variable. e.g.:

```
FOR X = 1:1:N  (range)
FOR Y = A,QZB, SQRT(Z),D,7,S  (list)
FOR Z = A,B,7.6,D:E:F,G,H  (both)
```

FOR can modify TYPE, PRINT, SET, DO, LINE, PAGE, SEND, PLOT, DELETE, DEMAND, READ, DUMP

IF — Command is executed if following comparison expression is true.

comparison operators

= < > <> <= >=

comparison subclause: expression-comparison operator-expression

```
A=B,  C<=D,  (A↑2+COS(B))>C[I,J]
```

Boolean operators

AND, OR, NOT

comparison expression: any combination of comparison subclauses and Boolean operators that has a unique true-or-false value.

IF clause with comparison expression

```
IF A=B
IF (A=B AND C<=D) OR F<>G
IF NOT (A=B OR C<=D) AND F<>G
```

IF can modify TYPE, PRINT, SET, DO, LINE, PAGE, SEND, PLOT, DELETE, DEMAND, READ, DUMP, LOAD, TO, DONE, STOP

MULTIPLE CLAUSES — Any meaningful combination of up to 6 FOR and IF clauses allowed. TELCOMP interprets multiple clauses from right to left. e.g.:

```
FOR A=1:1:N IF C<D FOR B=0:.04:M IF M<=1
```

IN FORM — Can modify TYPE and PRINT

```
TYPE A,B,C IN FORM 3
PRINT ALPHA IN FORM 1
```

FORM — A model line for typeout. Contains arbitrary text, fields. Fields allow exact specification of numerical typeout format. FORM numbers range from 1 to 99.

Decimal field: for number of fixed maximum size; typeout in decimal notation:

####, +#.###, -###.#

Exponential field: for any number within TELCOMP range; typeout in exponential notation:

##.#↑↑↑, +####↑↑↑

A FORM as entered by user:

TEMP ###.# PRESS. +#### PARTICLE VELOC. #.##↑↑↑

As typed by TELCOMP, numbers inserted:

TEMP 1.2 PRESS. -99 PARTICLE VELOC. 1.00+05

Can modify PLOT only. Provides X-axis scale. e.g.:

PLOT A ON B

Can modify DUMP only. Provides file name.

DUMP ALL PARTS AS ABCDEF

EDIT MODE

ESC key as command terminator or EDIT command causes TELCOMP to enter EDIT mode. TELCOMP retypes STEP or FORM if EDIT command given, retypes statement if ESC key was used before CARRIAGE RETURN.

To delete	Type \ under offending character
To replace	Type replacement under offending character
To append	Type any characters (including space) at end of line
To insert	Type ←. Ensuing characters are inserted before character above ←.
CTRL-TAB	Moves typehead to end of line. TELCOMP types intervening characters.
To terminate	Use ESC for verification or further editing, LINE FEED or CARRIAGE RETURN to enter edited material.

MATHEMATICAL ARGUMENTS

NUMBERS	Allowable range: 10^{-99} to 10^{+99} Accuracy: 9 significant digits Standard typeout: decimal notation for numbers with absolute value from .01 to 9999.9999. e.g.: 8765.43, -.0123, 5 Numbers outside the range above shown times a power of 10. e.g.: 1*10↑4, 1.456*10↑-6
VARIABLE	Up to 6 letters and digits, first of which is a letter (any combination not permanently defined in TELCOMP). e.g.: A, ABCDEF, A12345, ANSWER, AB
SUBSCRIPTS	Up to 4 allowed. May be denoted by number, variable, subscripted variable or expression. Allowable values 0 to 249. Subscripts are enclosed by square brackets. e.g.: A[77], A[0, 100, 200, 249], A[I,J,K], A[I[S[K,L], C+D*E]
EXPRESSION	Any combination of numbers, variables, algebraic operators and functions that has a numerical value. e.g.: A, 2+2, A↑2+D*X/COS (C)
ORDER	TELCOMP interprets multiple arguments from left to right.

SPECIAL SYMBOLS

$PI	π
$RN	Random integer between 1 and 131,071
$DATE	Present date. Eg: 2.0966*10↑4=2/9/66
$SEC	Present time (seconds from midnight)
$PG	Present page number
$L	Present line number
$BG	Storage blocks guaranteed (100)
$BU	Storage blocks used
$BA	Storage blocks available in common pool
$NBR	Channel number
$C	Line character position

CONTROL AND EDITING CHARACTERS

BREAK	Unconditionally returns control to the user
←	Printed by TELCOMP to indicate "YOUR TURN TO TYPE"
RUBOUT	Deletes all characters typed on present line
\	Deletes the last non-deleted character typed
;	Terminates a statement; further typing is a comment
,	Separates arguments in a list
" "	Used in TYPE and PRINT to output enclosed characters literally
RETURN	Statement terminator; enters statement
LINE FEED	Statement terminator; enters statement; supplies next STEP number if stored STEP was entered
ESC	TELCOMP enters EDIT mode, retypes last statement. If typed immediately after RUBOUT, restores deleted material.
CTRL-TAB	In EDIT mode, moves typehead to end of line.
CTRL-B	Same as ESC (Model 35 TTY)

APPENDIX VI

Section A: 1. List of Participating Schools and Teachers

2. List of Satellite Schools and Teachers

Section B: Map of School Locations and Computer Facility

Section A

1. Participating Schools and Teachers

ANDOVER

Crayton W. Bedford
Phillips Academy
Grade 9
Andover, Massachusetts

BELMONT

Edith P. Bixby
Payson Park School
Grade 6
Belmont, Massachusetts

Barbara J. Capron
J. S. Kendall School
Grade 6
Belmont, Massachusetts

Frances A. Conley
Winn Brook School
Grade 6
Belmont, Massachusetts

BROOKLINE

Clarence W. Bennett
Brookline Public Schools
Director of Mathematics
Brookline, Massachusetts

Phillip H. Johnson
Brookline High School
Grades 9, 10, and 11
Brookline, Massachusetts

Robert T. Wiggin
Brookline High School
Grade 11
Brookline, Massachusetts

LEXINGTON

John C. Dwyer
Lexington High School
Grade 9
Lexington, Massachusetts

Walter Koetke
Lexington High School
Grades 10 and 11
Lexington, Massachusetts

Neil Soule
Lexington High School
Grades 11 and 12 (Science)
Lexington, Massachusetts

WESTWOOD

Marcia N. Grey
Westwood High School
Grade 11
Westwood, Massachusetts

Mary Lou Liston
Westwood High School
Grade 11
Westwood, Massachusetts

Mary E. O'Malley
Westwood High School
Grade 11
Westwood, Massachusetts

James J. Pender, Jr.
Westwood High School
Grade 11
Westwood, Massachusetts

WINCHESTER

George C. Greer
Vinson-Owen School
Grade 6
Winchester, Massachusetts

2. Satellite Schools and Teachers

BRAINTREE

Robert P. Andersen
Thayer Academy
Grades 9 and 11
Braintree, Massachusetts

DEDHAM

Donald W. Seager
Dedham High School
Grades 10 and 11
Dedham, Massachusetts

DOVER

William M. Young
Dover-Sherborn Regional High School
Grades 11 and 12
Dover, Massachusetts

FRAMINGHAM

Joseph T. Hannigan
Framingham North High School
Grades 9, 10, 11, and 12
Framingham, Massachusetts

NEEDHAM

Frederick M. Cronin
Needham High School
Grades 10 and 11
Needham, Massachusetts

NEWTON

Nathaniel S. Merrill
Newton South High School
Grades 11 and 12
Newton, Massachusetts

Raymond D. Stephens, Jr.
Newton High School
Grades 10 and 12
Newton, Massachusetts

Herbert W. Waugh
Newton South High School
Grades 11 and 12
Newton, Massachusetts

NORWOOD

Louis J. Buscone, Jr.
Norwood Senior High School
Grades 10, 11, and 12
Norwood, Massachusetts

WALPOLE

Eleanor Radzwill
Walpole High School
Grade 11
Walpole, Massachusetts

WESTWOOD

Brother Roch
Xaverian Brothers High School
Grades 9, 10, and 11
Westwood, Massachusetts

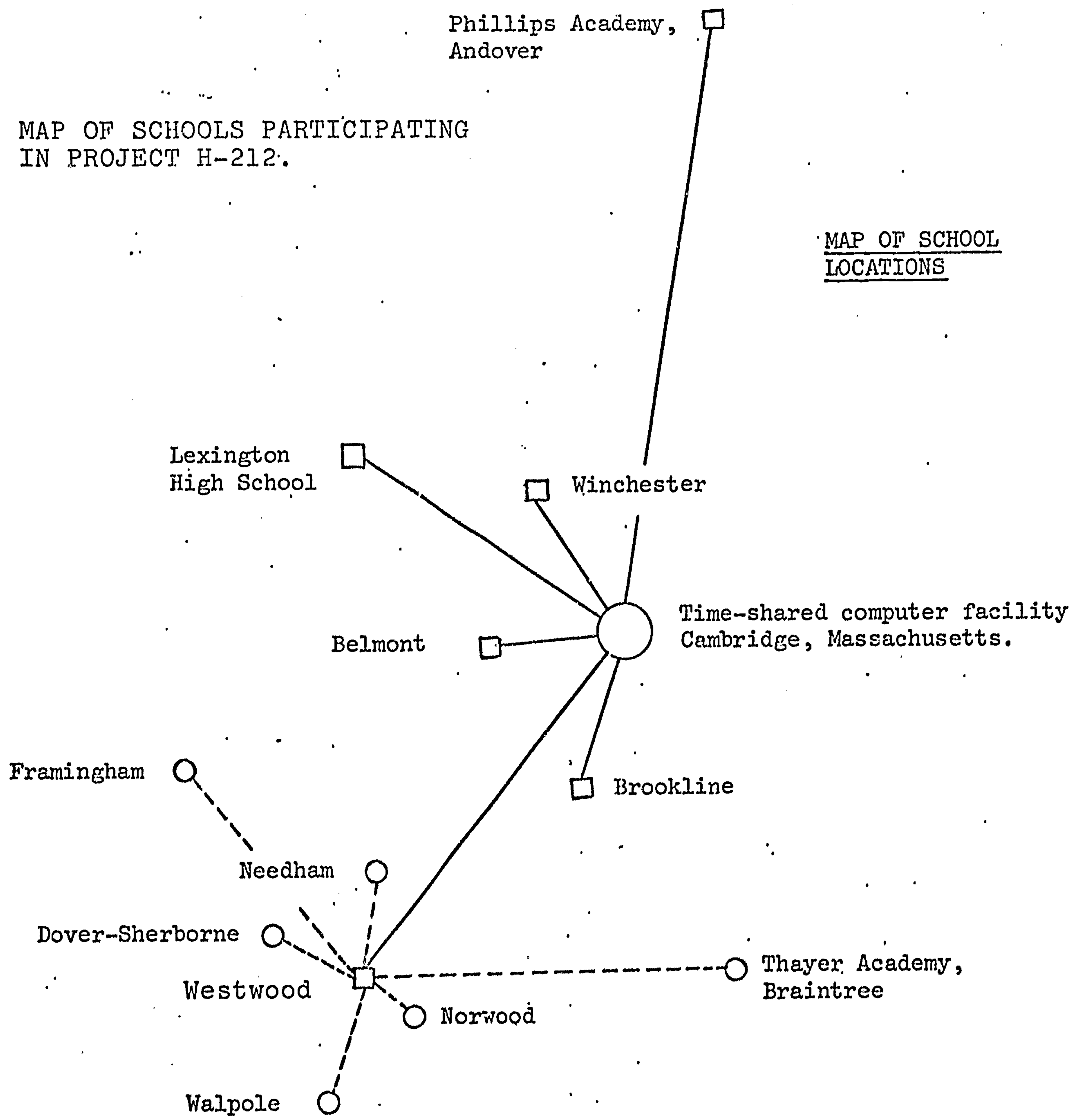
WINTHROP

James J. Vickerson, Jr.
Winthrop Schools
Head, Math Department
Winthrop, Massachusetts

APPENDIX VI

Section B. Map of School Locations and Computer Facility

MAP OF SCHOOLS PARTICIPATING IN PROJECT H-212.



□ Project schools; full-time use of terminals permanently located in school.

○ Satellite schools; part-time use of terminals.

——Direct teletype lines to cemtral computer.

---Teletype use by weekly trips to project schools.

Scale of Miles

0 5 10

APPENDIX VII

Section A. Advisory Board

PROJECT ADVISORY BOARD

G. Octo Barnett, M.D., Director
Laboratory of Computer Science
Massachusetts General Hospital
Boston, Mass.

Rev. Stanley J. Besuszka, S.J.
Director, Mathematics Institute
Boston College
Boston, Mass.

Dr. William Bossert
Computation Center
Harvard University
Cambridge, Mass.

Professor Robert Fano
Director, Project MAC
Mass. Institute of Technology
Cambridge, Mass.

Dr. Eugene Ferguson
Director of Mathematics
Newton Public Schools
Newton, Mass.

Mr. Hiram Haggett
Education Officer
Electronics Research Center
National Aeronautics and Space Administration
575 Technology Square
Cambridge, Mass.

Dr. Donald Hall
Professor of Science and Mathematics Education
School of Education
University of Massachusetts
Amherst, Mass.

Mr. Evan Herbert
Associate Editor
International Science and Technology
205 East 42nd Street
New York, N.Y.

Professor Anthony Oettinger
Chairman, Faculty Committee
Computation Center
Harvard University
Cambridge, Mass.

Dr. David A. Page, Director
University of Illinois Arithmetic Project
Educational Services, Inc.
108 Water Street
Watertown, Mass.

Professor Richard Pieters
Chairman, Mathematics Department
Phillips Academy
Andover, Mass.

Mr. Donald K. Pollock
Information Systems Branch
Office of Naval Research
Washington, D.C.

APPENDIX VIII

Consultants

APPENDIX VIII. Consultants

The research firm of Bolt Beranek and Newman, Inc., of Cambridge, Mass., provided scientific assistance to the Massachusetts Department of Education in carrying out the work of Project H-212(5-0311). Members of their Educational Technology Department, headed by Wallace Feurzeig, collaborated with Seymour A. Papert, Professor of Mathematics, The Massachusetts Institute of Technology, in initiating the investigation of a new approach to the presentation of mathematics based on the use of computers and programming languages. Other members of the staff of Bolt Beranek and Newman who provided consultative assistance are:

Daniel Bobrow	Staff Scientist
Geraldine Carey	Publications
D. Lucille Darley	Programmer
Robert Donaghey	Programmer
Richard Grant	Programmer
Richard Kahan	Staff Scientist
Vincent Sharkey	Staff Scientist
Cynthia Solomon	Programmer

Visiting Consultants

Sylvia Charp	Director of Computer Systems School District of Philadelphia
Lewis Clapp	Computer Research Corporation Newton, Mass.
Eugene Ferguson	Director of Mathematics Newton, Msaa., Public Schools
Glenn Culler	University of California Santa Barbara, California

In addition to the consultants listed above, members of the Advisory Board provided consultative assistance to the project staff from time to time during the research effort. (See Appendix VII for listing of Advisory Board)

OE 6000 (REV. 9-66)

DEPARTMENT OF HEALTH, EDUCATION, AND WELFARE
OFFICE OF EDUCATION

(TOP)

ERIC REPORT RESUME

ERIC ACCESSION NO.				
CLEARINGHOUSE ACCESSION NUMBER	RESUME DATE	P.A.	T.A.	
	06-01-68			

IS DOCUMENT COPYRIGHTED? YES ☒ NO ☐
ERIC REPRODUCTION RELEASE? YES ☒ NO ☐

TITLE

TEACHING MATHEMATICS THROUGH THE USE OF A TIME-SHARED COMPUTER

Final Report

PERSONAL AUTHOR(S)

Richardson, Jesse O.

INSTITUTION (SOURCE)

Commonwealth of Massachusetts, Department of Education; Boston, Mass.

SOURCE CODE

REPORT/SERIES NO.

OTHER SOURCE

SOURCE CODE

OTHER REPORT NO.

OTHER SOURCE

SOURCE CODE

OTHER REPORT NO.

PUB'L. DATE 29-Mar-68 CONTRACT/GRANT NUMBER OE-5-10-320 / 5-0311

PAGINATION, ETC.

218 pp.

RETRIEVAL TERMS

Teaching Mathematics through the use of a time-shared computer.
Use of a computer and programming languages (a) by individual students for independent study (b) as a mathematics laboratory to supplement classroom instruction.
Teaching programming languages as a conceptual and operational framework for the teaching of mathematics.

IDENTIFIERS

TELCOMP; Project H-212; Bolt Beranek and Newman, Inc.
Massachusetts Department of Education.

ABSTRACT

The work performed in Project H-212 sought to show that the teaching of the set of concepts related to computing, programming, and information processing could be used to facilitate and enhance the presentation of standard school mathematical curricular material, including arithmetic, algebra, and elementary calculus. Two new technological developments contributed significantly to the project effort. New mathematical languages had been developed to make computer programming very much easier to learn and to use in mathematical work. Also, time-shared multi-access computer systems with remote communication consoles were available, making the widespread use of computers in schools economically feasible. From the work carried out by the project at grades 6 through 12, the following conclusions are drawn; (1) It is possible to construct programming languages of great expressive power, yet simple enough to be taught to elementary school children. (2) Children are easily motivated to write programs at computer consoles; it is an enjoyable activity for them, regardless of their level of ability. (3)Children impose on themselves the for precision in thinking and expression in attempting to make the computer understand and perform their algorithms. (4) A series of key mathematical concepts such as variable, equation, function, algorithm, can be presented with exceptional clarity in the context of programming. (5) Computers and programming languages can be readily used as a mathematics laboratory facility to supplement regular classroom instruction.